TECHNICAL
REPORT

90-34

GEOTAB: User´s Guide
Version 1.8.2

Ergodata

October 1990

SVENSK KÄRNBRÄNSLEHANTERING AB
SWEDISH NUCLEAR FUEL AND WASTE MANAGEMENT CO

BOX 5864 S-102 48 STOCKHOLM

TEL 08-665 28 00 TELEX 13108 SKB S
TELEFAX 08-661 57 19

GEOTAB: USER'S GUIDE
VERSION 1.8.2

Ergodata

October 1990

This report concerns a study which was conducted for SKB. The conclusions and viewpoints presented in the report are those of the author(s) and do not necessarily coincide with those of the client.

Information on SKB technical reports from
1977-1978 (TR 121), 1979 (TR 79-28), 1980 (TR 80-26),
1981 (TR 81-17), 1982 (TR 82-28), 1983 (TR 83-77),
1984 (TR 85-01), 1985 (TR 85-20), 1986 (TR 86-31),
1987 (TR 87-33), 1988 (TR 88-32) and 1989 (TR 89-40)
is available through SKB.

GEOTAB: User's Guide

Version 1.8.2 23 Oct 1990

Design & Construction:

ErgoData

Contents

1. Introduction

This manual describes the general outlines, the everyday use and how to maintain and make changes in the Geodatabase system.

2. The Logon/Logoff Procedure

When you want to access the Geodatabase system you start by typing GEOTAB.

The program will present itself on your terminal and ask for your username and password to the Geodatabase. You type in these just as you did with the VMS counterparts.

```
***************************************************************

          ****                     *              *
         *    *    ****    ****   ****    ****    *
         *        *    *  *    *   *          *   ****
         *  ***   ******  *    *   *      *****   *   *
         *    *   *       *    *   *  *   *   *   *   *
          *****    ****    ****     **    ****** *****

 ***************************************************************

Type ? for help.
Type # for extended help.
Type ! to return to previous level.

                    Version 1.x

                    Username:
```

You may specify your username and password directly on the command line.

```
$ GEOTAB <username> <password>
```

The passwords are your keys to the computer and to the Geodatabase system. Do not lose them.

To exit from GEOTAB, type an "!" in the main menu. This brings you back to VMS "$".

3. The Menu System

All functions in the Geodatabase system can be reached through a series of menues. Each menu presents a number of alternatives and you select the desired alternative by entering a number and pressing return. The chosen alternative will lead you to:

- another menu
- a picture or series of pictures for a specific task
- a command interpreter where you can browse the database as far as your access rights permit.

Several keys have a special function for easy access to some often used function. These keys are valid throughout the Geodatabase system except where specifically noted in the following chapters.

”?” is used when you need help.

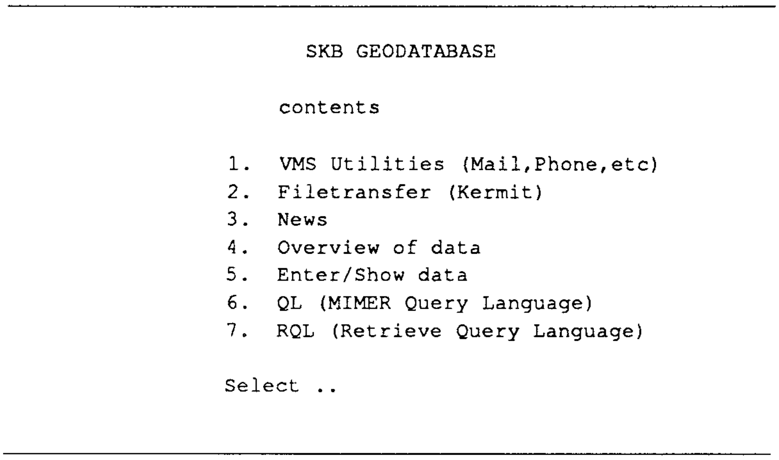
”$” gives you a list of valid alternatives.

”!” returns you to the previous level in the menu system. If you are at the top level, i e the main menu, you will be logged out from the GEOTAB program.

”CTRL–R” is used to enter a RQL command when you are working in another menu.

4. Main Menu

When you have entered the geodatabase system, by giving your username and password, you will be shown the main menu. The contents on the menu may differ depending on your privi-ledge level.



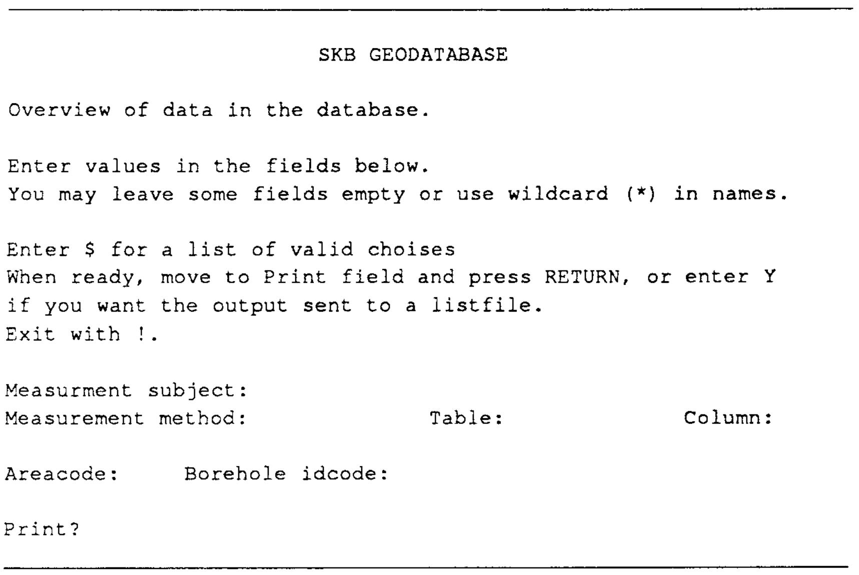
```
               SKB GEODATABASE

             contents

        1.   VMS Utilities (Mail,Phone,etc)
        2.   Filetransfer (Kermit)
        3.   News
        4.   Overview of data
        5.   Enter/Show data
        6.   QL (MIMER Query Language)
        7.   RQL (Retrieve Query Language)

        Select ..
```

The menuoptions available to the normal user are:

- VMS – show menu for VMS Utilities, such as Mail, Phone, etc.
- Filetransfer – transfer files between computers
- News – Show contents of the news file.
- Overview – Show methods used in certain areas or bore-holes
- Enter/show data – enter/show data using screenforms
- QL – database query language
- RQL – database query language

5. Overview of Data

The following picture will be shown on the terminal.



```
                              SKB GEODATABASE

Overview of data in the database.

Enter values in the fields below.
You may leave some fields empty or use wildcard (*) in names.

Enter $ for a list of valid choises
When ready, move to Print field and press RETURN, or enter Y
if you want the output sent to a listfile.
Exit with !.

Measurment subject:
Measurement method:            Table:            Column:

Areacode:      Borehole idcode:

Print?
```

The Overview facility enables you make a list of the current status of the amount of data entered into the database.

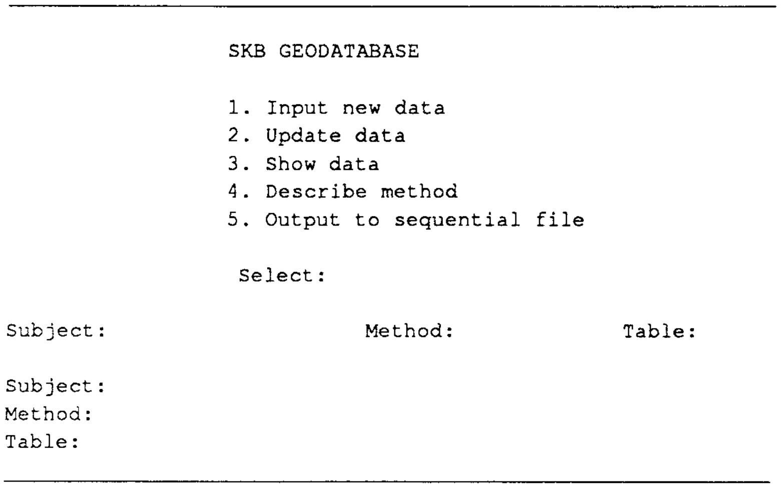
The $ character gives you a list of valid choises for each field. Select one from the list by typing the corresponding number.

Depending on which fields you enter data into, you will get different outputs.

Possible outputs:

Enter value for area.	List of all methods used in the area.
Enter value for subject and area– or idcode.	List of all methods from the selected subject that have been used in the area/borehole.
Enter value for subject, method.	List of all tables where data from the selected method have been entered.
Enter value for subject, method and table.	List of all columns in the table, min, max values and number of records.

6. Enter/Show Data



```
                  SKB GEODATABASE

                  1. Input new data
                  2. Update data
                  3. Show data
                  4. Describe method
                  5. Output to sequential file

                   Select:

Subject:                    Method:           Table:

Subject:
Method:
Table:
```

The program first asks for a menuchoice and then for a subject, a method and a tablename.

The subject is the type of measurement to which the method belongs (eg. hydro,chemi etc), a list of valid names will pop up if you enter a $ as the first character of the field. You may leave the subject blank if you wish.

Method is the name of the method you are interested in. The $ gives you a list of valid names. If you have selected a subject, only methods belonging to that subject are shown.

You can choose a certain table in the specified method by entering the table name or if you wish to start with the first table, just enter RETURN.

Alternative 1, 2 and 3 shows a serie of pictures, one for each flyleaf– and data–table in the method. Alt. 4 gives you a short description of the method and 5 asks for a sequetial file to dump the contents of the selected table in.

Only person responsible for entering data into the system have priveledge to use the insert an update alternatives (1 and 2).

6.1. Input New Data

The first picture for the selected method are shown, there is one field for each column in the table. On the last line on the screen there is a list of functions. You can execute them by entering the first character in the field after the list. This field is called the command field.

Possible commands in command field are:

GET_NEXT get next row from the current table. It is possible to limit the number of rows by entering values in one or more field before the first getnext command, see ENTER. If the preceding command was a GET_NEXT, it's not necessary to reenter the character, just hit RETURN and the command is repeated.

CLEAR clears all fields on the screen. The current table is not affected, only the screen.

ENTER moves the cursor from the command field to the first field of the picture. Use this command when you want to select one or a group of rows (records). If you enter a value in the last field, the cursor automatically moves to the command field. To get there from an other field use ! (exit character). All filled in fields are used as a select constraint in the following Get_next–commands, i e the contents of the table should match the contents of the fields.

INSERT inserts a new row (record) in the current table.

PRINT prints the current picture to the printfile.

TRANSACTION abort or commit the current transaction. A new transaction is automaticly started.

SEQUENTIAL_INPUT loads the current table with values from a sequential file.

NEXT_PICTURE shows the picture for the next table in the method. If the current picture is the last in the sequence, you are brought back to the Enter/show data menu.

> shows the picture for the next table in the method. If the current picture is the last in the sequence, you are brought back to the Enter/show data menu.

< shows the picture for the previous table in the method. If the current picture is the first in the sequence, you are not brought back to the Enter/show data menu.

! to leave the transaction. You will be asked if you want to abort (cancel all changes) the transaction or commit (keep all changes) it.

Valid exit characters in fields other than the command field:

! normal exit character, takes you back to the command field.

$ gives a list of valid values for the current field.

? displays the help text, if any, for the current field.

enters the Help Utility.

CTRL–R enter RQL–command.

CTRL–U gives a list of values stored in the column connected to the current field.

Valid exit characters in the command field:

! leave transaction.

? displays the help text for the command field.

CTRL–R enter RQL–command

Sequential Input

This command loads a table with data from a sequential file. The sequential file must contain all the primarykey–columns and be ordered one record/line.

Input files may be a sequential ascii file, or one of the following types: Symphony, dBase III+ or IV, or MIO.

After selecting file type, you enter the name of the file containing the data. If the file isn't in your current default directory, use full pathname.

Use ! to abort the transaction (brings you back to the command field).

When you have given a valid filename the following form appears.

```
         File specification for table <table_name>

=====================================================
column   start length  Icolumn   start length   Icolumn  start length
<col1>   <offs><len>   I                        I
<col2>   <offs><len>   I                        I
                       .                        .
                       .                        .
                       .                        .

                       I                        I
```

You can move among the fields with the arrow keys and change the values that don't match with your current file. If a column is missing in yor input file, or if you don't want to load the value, put a 0 (zero) in the length column for that field. When ready, put the cursor in the last field and press RETURN.

If you want to run the loading of data as a batchprocess, answer Y to the next question else N or RETURN and the loading of rows begins.

If you selected batch processing, a batch commandfile is created in your login–directory. The program asks if you want to put the batch in queue for immediate processing or not. If you answer N (no) you have to start the batch manually, see the description of the VMS SUBMIT command.

Diagnostic outputs during the load:

n rows loaded
this output appears for each 100th line.

record alread exist
there is already a record with the same primary key loaded in the table

illegal value: <value>
you tried to insert an illegal value in one of the columns.

n duplicates
the number of duplicate records found in the file

All messages are logged in a file named <filename>.LOG, where filename is the the name of the input file.

If one or more columns are missing in the input file, you can use Sequential_update to load them from an other file.

6.2. Update Data

The same pictures as for Input are used, but the list of commands are somewhat different.

Possible commands in command field are:

GET_NEXT
get next row from the current table. It is possible to limit the number of rows by entering values in one or more field before the first getnext command, see ENTER. If the preceding command was a GET_NEXT, it's not necessary to reeenter the character, just hit RETURN and the command is repeated.

CLEAR
clears all fields on the screen. The current table is not affected, only the screen.

ENTER
moves the cursor from the command field to the first field of the picture. Use this command when you want to select one or a group of rows (records). If you enter a value in the last field,

the cursor automatically moves to the command field. To get there from an other field use ! (exit character). All filled in fields are used as a select constraint in the following Get_next-commands, i e the contents of the table should match the contents of the fields.

MODIFY moves the cursor from the command field to the first field of the picture. First use the ENTER and GETNEXT commands to locate the data you want to change, then issue the MODIFY command and change the contents of the fields you want to update.

If you have change a value in a primary key field, a new record is inserted and the old record is deleted.

If you enter a value in the last field, the cursor automatically moves to the command field. To get there from an other field use ! (exit character).

INSERT inserts a new row (record) in the current table.

DELETE delete the current row (record) from the table. Used after a FIND or GETNEXT command.

PRINT prints the current picture to the printfile.

TRANSACTION abort or commit the current transaction. A new transaction is automaticly started.

SEQUENTIAL_UPDATE updates the current table with values from a sequential file.

NEXT_PICTURE shows the picture for the next table in the method. If the current picture is the last in the sequence, you are brought back to the Enter/show data menu.

> shows the picture for the next table in the method. If the current picture is the last in the sequence, you are brought back to the Enter/show data menu.

< shows the picture for the previous table in the method. If the current picture is the first in the sequence, you are not brought back to the Enter/show data menu.

HELP enter the Help Utility for Update_show.

! to leave the transaction. You will be asked if you want to abort (cancel all changes) the transaction or not.

Valid exit characters in fields other than the command field:

! normal exit character, takes you back to the command field.

$ givs a list of valid values for the current field. Arrow key right and left makes it possible to enter a searchpattern

? displays the help text, if any, for the current field.

enters the Help Utility.

CTRL–R enter RQL–command.

CTRL–U givs a list of values stored in the column connected to the current field.

Valid exit characters in the command field:

! leave transaction.

? displays the help text for the command field.

CTRL–R enter RQL–command

Sequential Update

This command could be used if you want to update one or more columns of a table with data from a sequential file. The sequential file must contain at least one primarykey–column and be ordered one record/line. It's also possible to make calculations on the values before update.

```
                              <table_name>

  1. <column1> <*>   2. <colummn2> <*>   3. <column3> <*>
     ........
     ........


                enter constant keycolumn/
                enter update column
columnnumber:..
```

1. Select the key–columns you want to declare as constant, i. e. keycolumns not present in the input file, or hit RETURN if none. For each selected column you are asked for a value.

2. Enter the columns that you want to update. RETURN when ready.

3. Do you want do make any calculations on the data?

If yes:

Enter the columns and the expressions to use. For valid expressions see documentaion on CALC.

4. Continue with field specification in the same way as for Sequential_input.

Diagnostic output during the update:

n rows updated

the number of rows currently updated

no row matched this line: <input line>

there was no record in the table matching the keys in this line

illegal value: <value>

you tried to insert an illegal value in one of the columns.

row not updated

you tried to store illegal data in a field or you don't have update privilege on the table

All messages are logged in the a file named <filename>.LOG, where filename is the name of the input file.

6.3. Show Data

The same pictures as for Input are used, but the list of commands are somewhat different.

Possible commands in command field are:

GET_NEXT

get next row from the current table. It is possible to limit the number of rows by entering values in one or more field before the first getnext command, see ENTER. If the preceding command was a GET_NEXT, it's not necessary to reenter the character, just hit RETURN and the command is repeated.

CLEAR

clears all fields on the screen. The current table is not affected, only the screen.

ENTER

moves the cursor from the command field to the first field of the picture. Use this command when you want to select one or a group of rows (records). If you enter a value in the last field, the cursor automatically moves to the command field. To get there from an other field use ! (exit character). All filled in fields are used as a select constraint in the following Get_next–commands, i e the contents of the table should match the contents of the fields.

PRINT

prints the current picture to the printfile.

NEXT_PICTURE

shows the picture for the next table in the method. If the current picture is the last in the sequence, you are brought back to the Enter/show data menu.

> shows the picture for the next table in the method. If the current picture is the last in the sequence, you are brought back to the Enter/show data menu.

< shows the picture for the previous table in the method. If the current picture is the first in the sequence, you are not brought back to the Enter/show data menu.

HELP enter the Help Utility for Update_show.

! to leave the transaction.

Valid exit characters in fields other than the command field:

! normal exit character, takes you back to the command field.

$ givs a list of valid values for the current field. Arrow key right and left makes it possible to enter a searchpattern

? displays the help text, if any, for the current field.

enters the Help Utility.

CTRL–R enter RQL–command.

CTRL–U givs a list of values stored in the column connected to the current field.

Valid exit characters in the command field:

! leave transaction.

? displays the help text for the command field.

CTRL–R enter RQL–command

6.4. Printing Output

The PRINT command in Input, Update and Show gives a copy of the screen. The output is placed in the current printfile. The file is opened on the first PRINT command and closed when leaving the transaction.

The name of the printfile and the printer–queue can be set by the RQL–commands

`SET PRFILE` and `SET PRINTER`.

You can use ctrl–r to issue the commands without leaving the transaction. The SET PRFILE must however be given before the first PRINT command.

If you want to close the printfile without leaving the transaction, use the RQL–command CLOSE PRINTFILE. The next PRINT command will open a new printfile.

7. RQL – Geotab Query Language

This part describes how to use the Retrieve Query and Command language (RQL).

The following form will be presented if you choose RQL from the main menu.

```
Type ! to return to menu

R>
```

RQL is a terminal based command language. This chapter is written with the implicit assumption that you work interactivly with RQL.

RQL takes input commands from a terminal, interprets them and executes operations on data stored in a the MIMER data base.

Syntax principles.

A command consists of key words, break characters (delimiters) and parameters.

The Key word identifies the command and forms its fixed parts. Examples of key words are DESCRIBE and SET.

The varaible part of the command is in the form of parameters such as databank names, table names or column names.

Key words and parameters are delimited by spaces.

When RQL expects a command, a promting symbol is displayed on the terminal. The promting symbol can be changed but has the default value R>. In the RQL command examples in this manual, the symbol used is R>.

A command may stretch over several lines. For each new line you are prompted with the continuation prompter +>. Each command may be terminated by a ; or an empty line.

Example:

```
R> DESCRIBE TABLE table-name;
```

or

```
R> DESCRIBE TABLE table-name
+>
```

Between key words, parameters and break characters an arbitrary number of spaces are allowed.

The key word may be shortend down as long as it is nonambiguous.

The question mark (?), could be used instead of a keyword. A list of possible keywords is then dispayed.

Example:

```
R> DESCRIBE ?;
```

would produce the following list:

```
DOMAIN     Describe domain
TABLE      Describe table
COLUMN     Describe column
METHOD     Describe method
TEMPTABLE  Describe temptable
```

Some of the settings made by the RQL–command SET are also valid in other transactions. These are:

```
SET PRInter
SET PRFile
SET PRHead
SET PRRow
SET PRCol
SET SCWidth
SET SCRoll
SET Missingval
```

These commands may also be given in the initfile (geotab.ini).

7.1. RQL COMMANDS

The available commands are:

CALC	Make a calculation.
CHANGE LOGIN	Change the login ident.
CLOSE	Close a log or printfile.
COMMAND	Sends the rest of the string to the VMS commandinterpreter.
COORDINATES	Calculates the coordinates for a specified length along a borehole.
COUNT	Count rows in a table
DELETE	Delete temptable
DESCRIBE	Describes different objects in the database.
HELP	Enter RQL help utility.
MANUAL	Make a manual.
PRINT	Print output from RQL–command to a sequential file.
QL	Sends the rest of the string to the MIMER/QL command interpreter.
SELECT	Select data from one table, or two tables (join).
SET	Set environment variables.
SHOW	Show environment variables.
TIME	Show current time
TRANSFER	Transfer a file from VAX to PC

CALC Command

Make calculations. You can enter an expression on the command line and calc will give you the answer directly or you can enter the CALC utility by typing only the command on the commandline. See Calc a Calculator Program in the Calc chapter for more information.

Syntax CAlc [expression]

or

CAlc

The latter form is used to enter interactive CALC.

CHANGELOGIN Command

Change to another ident. To get back to the previous ident, issue a new changelogin command.

Syntax CHangelogin [ident]

You will be asked for a password. If not accepted you will be left at current loginident. You may only use CHANGELOGIN to change to a user ident.

COMMAND Command

This command sends the rest of the commandline to VMS for processing.

Syntax COMmand string

or

COMmmand

The latter form is used to enter interactive DCL.

Example:

```
R> COMmand DIRECTORY *.DAT
```

for a list of all .DAT files in your current directory.

```
R> COMmand;

$ <enter some dclcommands>
...
$ logout        !leave interactive DCL
R>
```

COORDINATES Command

Show local coordinates for the specified length along the borehole, measured from top of casing

Syntax COOrdinates [[idcode] [length along borehole]]

COUNT Command

The count command count rows in a table.

Syntax COUnt [[creator].[table_name]]

DELETE Command

The delete command delete temporary tables

Syntax DELete temptable table_name

DESCRIBE Command

The describe command describes an object.

Syntax DEScribe Column [[creator].[column_name]]
DEScribe Domain [[creator].[domain_name]]
DEScribe Table [[creator].[table_name]]
DEScribe Method [method_name]
DEScribe Temptable [temptable_name]

The format of the commands are:

```
R>DESCRIBE TABLE table-name
```

Wildcards are allowed in table–name. A description of the table columns is listed at the terminal.

```
R>DESCRIBE METHOD method-name
```

Wildcards are allowed in method–name. A description of all tables in the method is listed at the terminal.

```
R>DESCRIBE COLUMN column-name
```

Wildcards are allowed in column–name. For each table where the column is found, a description of the table and the column is displayed.

```
R>DESCRIBE DOMAIN domain-name
```

Wildcards are allowed in domain–name. A description of the domains is listed at the terminal. Information about domain length, type and columns connected to the domain are displayed on the terminal.

```
R>DESCRIBE TEMPTABLE
```

Wildcards are allowed in temptablename. A description of all available temptable created by your VMS login ident is listed at terminal.

HELP

The HELP command displays the help topics for RQL. The command uses VMS Help Utility. Use the questionmark (?) in response to the Topic? or Subtopic? prompts to redisplay the list of topics available at the current level. For more information about the Help Utility, refer to the VMS Utility manual.

Syntax Help [command]

MANUAL Command

The manual command makes a manual with method and table describtions.

Syntax MAnual Method [[creator].[table_name] [application]]
MAnual Domain [creator].[domain_name]

PRINT Command.

The PRINT command will put the output from the commands stated in the PRINT command in the current printfile. When the file is closed you are asked if it should be printed. The default value for printer ident is SYS$PRINT and for the printfile SKBPRINT.LIS. This can be changed by the SET PRINTER command and the SET PRFILE command.

Syntax Print RQL–command

where RQL–command is one of:

DESCRIBE
SELECT
COUNT

QL Command

The QL–command will send the rest of the line to MIMER/QL for processing. A PRINT command in front of a QL command will have no effect. To get a printout from QL–commands, use the syntax stated in the QL manual.

Syntax QL string

SELECT Command

Syntax SELect [[creator].table1 [[creator].table2]]

```
           table1                          [table2]
col1       ....                    [col1]
col2       ....                    [col2]
col3                               [...]
...
...
----------------------------------------------------------------



                    select column:.........
```

All columns in the table(s) are listed on the terminal and you can select the columns you want to see, by typing the corresponding number, a range (e.g. 1–5), a list (e.g. 6,8,11) or a * if you want all columns. It is also possible to use the colonsyntax (e.g. :2 – :8,:11).

Calculation In Search Process

If you want do add a calculated column, enter a + as column namn, and give column heading and calculation expression. You can include other column values in the expression by entering the column name or number (:number). It is only possible to make calculations on values from the current row. For valid expressions, see CALC.

Calculated columns can also be used if you want to change name on a column, in that case, enter the real columnname as expression.

Condition Clause In Select Command

When you are ready, enter an empty line as column–number. You will then be prompted for optional conditions. If you are joining two tables, you must enter at least one connection between the tables, normally between fields belonging to the same domain. Enter columnreferences as columnname or columnnumber preceded by a colon (:).

Available operators are:

=	equal
<>	not equal
>	greater than
>=	greater than or equal
<	less than
<=	less than or equal
between .. and ..	range
not between .. and ..	range

Example:

between 20 and 30

which can be applied to both numerical and character columns and

LIKE	string matching pattern
NOT LIKE	string not matching pattern
CO	contain
BW	begin with
NB	not begin with
NC	not contain

The following operators are equal:

CO = LIKE *string*
BW = LIKE string*
NB = NOT LIKE string*
NC = NOT LIKE *string*

which are valid only for character columns only, and the logical operators

AND	logical and

OR logical or

Finish with an empty line.

Then select the output destination, the terminal, a temporary table or a file. The program can produce outputs

in several different fileformats. You will get a list of choises when you select formatted output.

Select From One Table

Example:

```
                 table1
        1* col1  4. col4  7. col7
        2* col2  5. col5
        3. col3  6. col6
---------------------------------------------------------
col1    col2   col6   col7


where :1 = ABC and :2 > 5
```

Select all rows where column1 equals the string 'ABC' and column is greater than 5. Show column 1,2,6 and 7.

Select From Two Tables

Example join:

```
        table1                        table2
1* col1  4. col4  7. col7      8* col8   11. col11
2* col2  5. col5               9. col9
3. col3  6. col6              10. col10
---------------------------------------------------------
col1    col2   col8   col11


where :8 = :1 and :2 > 5
```

Select all rows from table1 where column2 > 5 and for each row found, find all rows in table2 where column8 (the first column of table2) equals the current value of column1. Show column 1, 2, 8 and 11.

Outer_Join

Outer join can be chosen when typing ~ (tilde) as the first character on the condition line. Your keyboard may have a ü keytop instead of ~.

The outer join is used to indicate that the table which may be missing entries shall be treated as though it had an additional (empty) row which will be joined to any row of the other table which would be otherwise discarded.

An outer join will return those rows that have no matching in the join case.

Example outer join:

```
          table1                            table2
1* col1  4. col4  7. col7      8* col8   11. col11
2* col2  5. col5               9. col9
3. col3  6. col6              10. col10
----------------------------------------------------------
col1     col2    col8    col11


where ~:8 = :1
```

Select all rows from table1 and for each row, find all not existing rows in table2. Show column 1, 2, 8 and 11.

Interpolation

If you want to join two tables with nonmatching keyvalues, e. g. the first table has values for 10,20,30 etc. and second has values for 5,15,25 etc. , it is not possible to use standard join.

To be able to join such tables, the values in one of them have to be interpolated. This function is available by substituting the equality sign '=' by the interpolation sign '#' in the where clause.

The values in the second table in a join are interpolated to match the values in the first table. The interpolation column must belong to the primary key.

Example:

```
where col1 # col8 and col1 = 'KKM02'
```

Missing Values

If you want to select (or exclude) rows, where a column has no value, use the identifier :NULL.

Example:

```
where col1 <> :NULL
```

excludes all rows where col1 has a missing value

SET Command

Set environment varibles.

Syntax

SET Directory [directory_name]
SET Columnsep [value]
SET Log <ON|OFF>
SET Menu <ON|OFF>
SET Missing [value]
SET PRInter [printer_name]
SET PRFile [printer_filename]
SET PRHead <ON|OFF>
SET PRRow [value]
SET PRCol [value]
SET PROmpt [prompt_string]
SET Reportint
SET SCRoll <ON|OFF>
SET SCWidth [value]

```
R> SET Directory [directoryname]
```

Sets the current directory to another directory. The output from geotab will be stored on this new directory. At logout from geotab you will return to old current directory.

```
R> SET Columnsep [value]
```

The set column separation command sets the column separation sign to another sign than default.

```
R> SET Missing [value]
```

Set missing value sets missing values to other than default.

```
R> SET LOG <ON|OFF>
```

Starts/stops the session logging. The output from the RQL–session will be stored in a logfile. You are prompted for the name of the file. The output is not always identical with the output to the screen, but the contents should be the same.

```
R> SET MENU <ON|OFF>
```

Begin or end menu mode. In menu mode, all possible command are shown in a menu. You select a command by typing the corresponing number. If a command consists of more than one keyword (e.g. SET), a new menu with possible subcommand appears after selection of such commands.

```
R> SET PRINTER printername
```

Enables you to change the name of the printing device. Default is SYS$PRINT. Consult your system manager to get the name of the printer at your site. This setting is valid in all transactions using printout.

```
R> SET PRFILE printfilename
```

Enables you to change the name of the printfile. Default is SKBPRINT.LIS. This setting is valid in all transactions using printfiles.

```
R> SET PRHEAD <ON|OFF>
```

Enable or disable printing of columnheading on top of each page in the printerfile. (Output from SELECT).

```
R> SET PRROW [value]
```

Set number of rows/page in the printerfile. Value = 0 indicates that there will not be any paging.

```
R> SET PRRCOL [value]
```

Set number of characters/line in the printerfile.

```
R> SET PROMPT promptstring
```

Enables you to change the default prompt in RQL

```
R> SET SCROLL <ON|OFF>
```

The scroll flag controlls the output from several RQL commands. With scroll on, output will not stop at each screenpage. Default is scroll off.

```
R> SET SCWIDTH [value]
```

The screenwidth for output from SELECT is set to [value]. The width can be set between 20 and 200 characters / line.

```
R> SET Reportint [value]
```

Sets the interval for reports from select and load on number of rows found or loaded.

SHOW Command

The show command shows the current value of some parameters.

Syntax

```
SHow Settings
SHow temptable
SHow time
```

TIME Command

Shows current time in a human readable form.

Syntax

```
TIme
```

Output examples:

```
Quarter to five
Half past two
```

TRANSFER Command

The command can be used to send files from the VAX to your local computer (PC). You must run a program capable of the Kermit filetransfer protocol. If you are using VTKermit on a IBM–PC or compatible, you only need to enter the Transfer command and the file will be transferred. On other impementations of Kermit you must exit back to command level on your local Kermit and issue a Receive command, after entering the Transfer command.

Syntax

```
TRansfer [filename]
```

7.2. INITFILE

When you first log on GEOTAB, the program looks for a file named GEOTAB.INI in your current default directory and your login directory. This file may contain valid RQL commands. The file is read and the commands executed after a successful login, but befor the first menu appears on the screen. The main purpose of the initfile is to allow setting of environment variables such as printername, lines/pages etc.

Ex. Rql commands in the initfile:

```
Set printer txa0;
Set prcol 128;
Set prfile MYFILE.OWN;
```

This will direct output to MYFILE.OWN instead of the default SKBPRINT.LIS, allow 128 characters per line and use TXA0 as default printer.

7.3. Batch Directives

It's possible to run Geotab as batch process. The commands in batch mode differs somewhat from interactive mode. Here is an example commandfile (RQLTEST.COM):

```
$ geotab <username> <password>      - DCL command
rql;                                - choose rql from main menu
select bhcoord;                     - select table bhcoord
idcode                              - select columns
xcoord
ycoord
zcoord
                                    - finish selection with empty line
idcode = kkm02;                     - selection criteria
f                                   - output destination
SYMPHONY                            - Symphony conversion
kkmsymph                            - filename for converted output
                                    - one empty line
!;                                  - leave rql
!;                                  - leave geotab
$
```

All menuselections must be entered explicitly. Ex. rql; or SYMPHONY. All other commands should be given exactly as in interactive mode.

Submit to batch queue with the command:

```
SUBMIT RQLTEST.COM
```

For more details on starting a batch process, see VMS SUBMIT command.

8. CALC – A Calculator Program

Calc is a program for mathematical calculations for which you might use a hand held calculator. calc supplies most of the operations common to programming languages and variables with properties much like those in Visicalc.

The arithmetical operators calc offers are

+	addition
–	subtraction and change–sign
*	multiplication
/	division
%	modulo division
^	exponentiation

Arithmetical expressions can be arbitrarily complex and are generally evaluated left to right. That is,

a + b – c

is the same as

(a + b) – c.

Exponentiation is evaluated before multiplication and division which are evaluated before addition and subtraction. For example, the expression

a + b – c * d / e ^ 2

is parsed as

(a + b) – ((c * d) / (e ^ 2))

This default order of operations can be overridden by using parentheses.

Calc supplies some transcendental functions: sqrt, log, exp, and abs, and the following trigonometric functions: sin, asin, cos, acos, tan, and atan, for which degrees are measured in radians.

8.1. Using CALC

To use calc, begin by typing calc at the RQL command level, and calc will prompt you with

```
CALC:
```

Type in each of your expressions followed by RETURN and calc will respond with how it parsed your expression followed by the result. In all following examples, what you would type in is preceded by the calc prompt

```
CALC:
```

and what calc responds with is immediately after. A simple calculation is:

```
CALC: sqrt (12^2 + 5^2)
sqrt(((12 ^ 2) + (5 ^ 2)))     = 13
```

Expressions can be stored by assigning variables to them. For example you could type:

```
CALC: pi = 22/7
(22 / 7)      = 3.14286
CALC: pi
pi          = 3.14286
```

Variables can be used in expressions.

```
CALC: area = pi * r^2
(pi * (r ^ 2))      = UNDEFINED
CALC: area
area     = UNDEFINED
```

area is undefined because r has not been set. Once r is set, area will have a value because area is set to an equation rather than a particular value. This can be observed by printing all the variables so far introduced with ^V (CTRL–v), which may have to be typed twice as ^V is used to quote characters.

```
CALC: ^V
pi       =       3.14286 = (22 / 7)
area     =     UNDEFINED = (pi * (r ^ 2))
r        =     UNDEFINED =
```

The variable table is formatted so that each variable's name is on the left, followed by its current value, followed by its current definition. If r is set to 5, the value of area is now defined.

```
CALC: r = 5
5         = 5
CALC: ^V
pi       =       3.14286 = (22 / 7)
area     =       78.5714 = (pi * (r ^ 2))
r        =             5 = 5
```

The effect of changing r on area can be easily observed because of the way area is defined.

```
CALC: r = 2
2         = 2
CALC: area
area      = 12.5714
```

8.2. Setting Constant Values

Of course, there are times when you want to set a variable to a value and not have it depend on the values of variables at a later time. To do this, you precede an expression with the number operator #. For example,

```
CALC: area2 = # area

12.5716       = 12.5716
CALC: ^V
pi        =      3.14286 = (22 / 7)
area      =      12.5716 = (pi * (r ^ 2))
r         =            2 = 2
area2     =      12.5716 = 12.5716
```

area2 does not depend on the variable to which it was set because the number operator only lets numbers through it rather than expressions. If area2 was set without the operator, it would be subject to any changes in area or to any changes in variables on which area depends.

```
CALC: area2 = area
area      = 12.5716
CALC: ^V
pi        =      3.14286 = (22 / 7)
area      =      12.5716 = (pi * (r ^ 2))
r         =            2 = 2
area2     =      12.5716 = area
```

8.3. Testing Conditions

Variables can be set based on a tested condition. For example, you may want a variable max to always be the maximum of a and b.

```
CALC: max = if a > b then a else b
(if (a > b) then a else b)     = UNDEFINED
```

max is undefined because a and b have not been set.

```
CALC: a = 21
21      = 21
CALC: b = 3^3
(3 ^ 3)     = 27
CALC: max
max       = 27
CALC: a = 50
50    = 50
CALC: max
max       = 50
```

The if–then–else expression allows variables to be set based on conditions. This condition can be made up with relational and logical operators. The relational operators available with calc are:

==	test equality
!=	test inequality
>=	greater than or equal
<=	less than or equal
>	greater than
<	less than

while the logical operators are:

&	and
|	or
!	not

A more complicated expression involving these is:

```
if a > b & b > c then b
```

The else part of the conditional is optional, and if not present and the condition is false, the conditional is undefined.

8.4. Undefined Variables

Variables are undefined if they have not been set, if they depend on variables that are undefined, or if they are set to an expression involving an illegal operation.

```
CALC: 1/0
(1 / 0)       = UNDEFINED
```

You can be confident that no operations will result in calc blowing up. Thus you could write the equation for the roots of a quadratic formula with the following definitions and always get reasonable answers.

```
x = 0
a = b = 1
c = -1
radical = sqrt (b^2 - 4*a*c)
equation = a*x^2 + b*x + c
derivative = 2*a*x + b
root1 = (-b + radical) / (2 * a)
root2 = (-b - radical) / (2 * a)
```

8.5. Control Characters

Non–mathematical operations are accomplished with control characters. To type a control character, say CTRL–p, while you hold down the key labeled CTRL you type a p. This will appear as ^P. Some control characters have very special meanings, such as "stop the program" so you must be careful with them. In general, you can avoid any problems with control characters by typing a ^V (CTRL–v) before them. This character removes any special meaning associated with the character immediately following it. So to type ^P you could be extra safe and type ^V^P. To type a CTRL–v, you may have to type it twice. Unfortunately, these conventions are not universal.

The following control operations are available with calc.

^P	change the printing option
^Gf	read the input from file f and return to current state
^V	print the variable table
^Wf	write the variable table to file f

If you forget any of these commands, you can type a ? to get calc to remind you.

8.6. Operations

OPERATOR		*ASSOCIATIVITY*	
	PRECEDENCE		*DESCRIPTION*
#a	1	none	numerical value of a
a=b	2	right	a is set to expression b
if a then b	3	left	if a != 0 then b else UNDEFINED
else	4	left	
a|b	5	left	true if a or b is true
a&b	6	left	rue is a and b are true
!a	7	none	true is a is false
a==b	8	none	true if #a equals #b
a!=b	8	none	true if #a is not equal #b
a<b	8	none	true if #a is less than #b
a>b	8	none	true if #a greater than #b
a>=b	8	none	true if #a > #b | #a == b
a<=b	8	none	true if #a < #b | #a == b
a+b	9	left	a plus b
a–b	9	left	a minus b
a*b	10	left	a times b
a/b	10	left	a divided by b
a%b	10	left	a modulo b
a^b	11	right	a to the b
–a	12	none	change sign
abs(a)	12	none	absolute value
exp(a)	12	none	e to the a
log(a)	12	none	10 – logarithm of a
sqrt(a)	12	none	square root of a
sin(a)	12	none	sine of a in radians (cos & tan)
asin(a)	12	none	arc sine of a (acos & atan)

9. QL – MIMER Query Language

Included in the Geodatabase system are also MIMER standard query language QL. This language enables you search freely trough the all tables in the database.

This information has been extracted from the MIMER/QL Users Manual from Mimer Information Systems.

MIMER/QL is used in combination with the database handler MIMER/DB to define and build up databases as well as to search for and manipulate data. QL is an abbreviation for query language. The concept query language refers to the possibility, with the help of the language, of asking spontaneous, ad–hoc questions, and getting an answer about the contents of a database. However, MIMER/QL can be used for more advanced tasks than this, for example, for creating systems based entirely on so called menus, or for doing calculations.

The following form will be presented if you choose QL a query language in the main menu.

```
MIMER QUERY LANGUAGE
Type EXIT; to return to menu

QL>
```

9.1. QL Commands

The available commands are:

ALIAS

Defines one or several aliases for table name

Syntax
ALIas table–name (alias,...)<,...>;

COMMAND

Activates special computer system dependent commands tailored to a specific implementation.

Syntax
COMmand 'string'

COPY ... FROM

Copies data from a sequential file to a MIMER table

Syntax

```
COPy </INSert /LOAD> >/DLOgg /NODlogg> tabref
FROm <’>file_name IS format,...)>;
```

COPY ... TO

Prints rows from a MIMER table on a sequential file

Syntax

```
COPy tabref TO <’>file_name<’> <( <nX,> column_name
               IS format,...)>;
```

DBA

Invokes the Data Dictionary Service Utility for database definition operations DDS is a synonymous command

Syntax

```
DBA;
```

DDS

Invokes the Data Dictionary Service Utility for database definition operations DBA is a synonymous command

Syntax

```
DDS;
```

DEFINE INDEX

Creates an index for column names in the table name

Syntax

```
DEFine INDex_name ON table_name (column_name);
```

DEFINE TABLE

Defines a MIMER table

Syntax

```
DEFine TABle table_name ( column_name IS format
               <:column_name IS format>
               <,column_name IS format> )
               IN databank_name;
```

DELETE

Deletes data from a whole table or (with a condition clause) one or several rows from a table

Syntax

```
DELete tabref <WHEre condition clause>;
```

DESCRIBE ALIAS

Lists all aliases defined during the terminal session

Syntax

```
DEScribe ALIas <table_name>;
```

DESCRIBE AREA

Shows all tables opened during a QL session

Syntax

```
DEScribe AREa;
```

DESCRIBE TABLE

Shows all tables opened during a QL session

Syntax

DEScribe </T /P /A> </BRIef ALL> TABle_name;

DO

Reads a sequential file and reads its lines as /QL commands

Syntax
DO <'>file_name<'>;

EDIT

Activates the /QL procedure editor

Syntax
EDIt <procedure_library> <(procedure)>;

ENTER

Gains access to the privileges granted to a program

Syntax
ENTer program_name <<'> PASSWORD <'>>;

EXECUTE

Starts a pre–defined /QL procedure

Syntax
EXEcute <procedure_library> <(procedure_name)>;

EXIT

Leaves /QL and stores all tables in their current status

Syntax
EXIt;

GET

Retrieves data from one or more tables: the search of specific rows is governed by a condition clause

Syntax
GET </T /P /A /N> target list
<WHEre condition_clause> <order_clause>;

or

GET table_name (target_list) <WHEre condition clause>;

HELP

Lists available commands or with operand the command's syntax. In error message environment lists help text

Syntax
HELp <command error_code>;

INSERT

Inserts rows in specified table columns or in promted table columns

Syntax
INSert tabref;

or

INSert tabref (column_name = <'> data >'>);

LEAVE

Resumes the ident preceding an ENTER command

Syntax
LEAve;

PRINT

PRInt < /NORepeat <(column_name,...)> table_name <'Headline–text'> <order_clause>;

REDEFINE

Changes the definition of an empty table

Syntax
REDefine table_name
APPend (column_name IS format,...);

REMOVE INDEX

Removes a defined index

Syntax
REMove INDex index_name;

REMOVE TABLE

Eliminates the data and the definition of a table

Syntax
REMove TABle table_name;

RESET

Resets all mode variables to standard values, closes printfiles and eliminates all aliases or resets a specified mode variable to its standard value

Syntax

```
RESet  < CC     > < ,Echo > < ,FLEnm > < ,Gag >
<,ILEnm   > < ,LC   > < ,Long  > < ,LW  >
<,PL      > < ,PLS  > < ,PRIntf> < ,PW  >
<,PROclib > < ,Trace> < ,Verify> < ,WORkdb> ;
```

SET

Changes a standard mode variable be operand

Syntax

```
SET          < CC = n >
             <,ECHo  NOEcho >
             <,FLEnm = FLENCm >
             <,GAG   NOGag  >
             <,ILEnm = ILENCm >
             <,LC  = n      >
             <,LONg   Short  >
```

```
<,LW  = n     >
<,PL   = n     >
<,PLS = n     >
<,PRIntf = 'file_name' >
<,PW  = n     >
<,PROclib = Procedure_libary >
<,TRAce  NOTTrace >
<,VERify    NOVerify >
<,WORkdb = Databank_name >;
```

SHOW   Shows at the terminal – depending on operands – current session's aliases, data buffers, databanks, tables, settings present time, version and logged–on users

Syntax

```
SHOw      ALIas <table_name>
          BUFfers
          INCluded
          SETtings
          TIME
          USEr
          VERsion;
```

UPDATE   Replaces data in specified non–primary columns where the rows are defined by a condition

Syntax

```
UPDate tabref (column_name = data, ...)
<WHEre condion_clause>;
```

Appendices

A. File Transfer

When you chose Filetransfer in the main menu the following form will be presented for you.

```
Type EXIT to return to menu
VMS Kermit-32 version X.X.XXX
Default terminal for transfer is : XXXX:
Kermit-32>
```

The problems to transfer files between computers can be solved in many ways. One way is cheap and relatively easy. Connect the two computers through their terminal ports (TTY), tricking one computer (or both) into believing that the other is a terminal. Once two computers are connected in this way, cooperating programs can be run on each to achieve the desired communication by means of a communication protocol.

A protocol is necessary to prevent corruption of data and to synchronize communication, cooperating computers can send control information to one another at the same time that they are transferring data. This intermingling of control information with data, and the resulting actions, constitute a protocol.

KERMIT is such a protocol. It is specifically designed for transfer of sequential files over ordinary serial telecommunication lines. KERMIT is not necessarily better than many other terminal–oriented file transfer protocols but it is free, it is well documented, and it has been implemented compatibly on a variety of microcomputers and mainframes.

KERMIT transfers data by encapsulating it in packets of control information. This information includes a synchronization marker, a packet number to allow detection of lost packets, a length indicator, and a checksum to allow verification of the data. Lost or corrupt packets are detected, and retransmission is requested. Duplicated packets are discarded. In addition, various special control packets allow cooperating KERMITs to connect and disconnect from each other and to exchange various kinds of information.

Available commands in Kermit–32 for filetransfer are:

SEND filespec Send the file or filegroup specified by filespec from this Kermit the other.

RECEIVE Receive a file or filegroup from the other Kermit.

SERVER Act as a server to a remote Kermit.

CONNECT Make a virtual terminal connection to the remote system.

SET Establish various nonstandard settings, such as connect ascape character, file characterisics, communication line number, parity, or flow control.

SHOW Display the values of SET options

HELP Type a summary of Kermit commands and what they do.

EXIT EXIT from Kermit back to the host operating system.

B. RQL Examples

B.1. RQL COMMANDS

The available commands are:

CALC	Make a calculation.
CHANGE LOGIN	Change the login ident.
CLOSE	Close a log or printfile.
COMMAND	Sends the rest of the string to the VMS command interpreter.
COUNT	Count rows in a table
DELETE	Delete temptable
DESCRIBE	Describes different objects in the database.
HELP	Enter RQL help utility.
MANUAL	Make a manual
PRINT	Print output from RQL–command to a sequential file
QL	Sends the rest of the string to the MIMER/QL command interpreter.
SELECT	Select data from one table, or two tables (join).
SET	Set environment variables.
SHOW	Show environment variables.
TIME	Show current time
TRANSFER	Transfer a file from VAX to PC

SELECT Command

The select command selects data from one table, or two tables (join)

Syntax SELect [[creator].table1 [[creator].table2]]

Example – Select Table

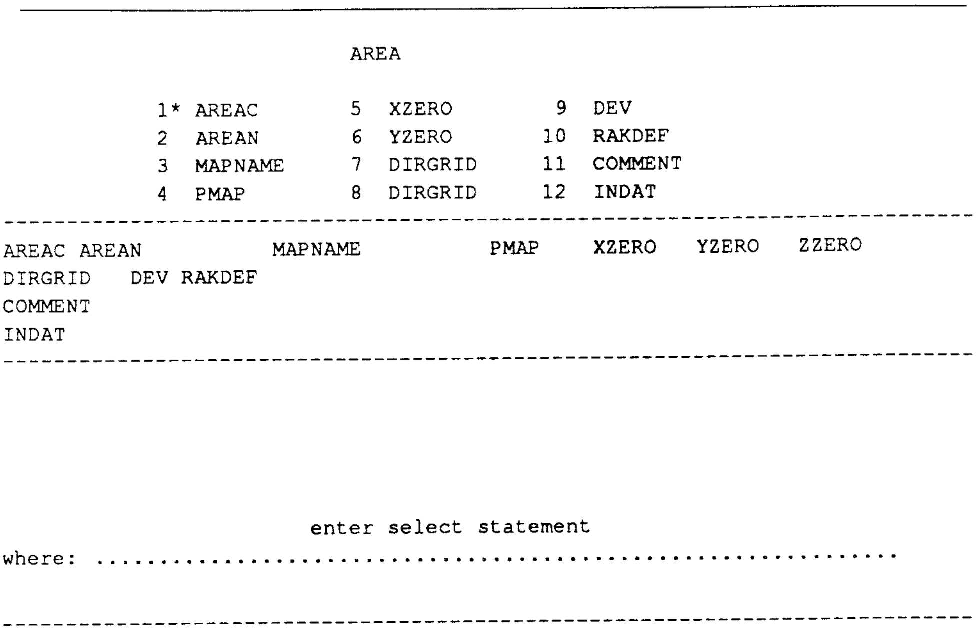
```
R> SELECT AREA;
```

```
                                  AREA

                    1* AREAC      5  XZERO      9  DEV
                    2  AREAN      6  YZERO     10  RAKDEF
                    3  MAPNAME    7  DIRGRID   11  COMMENT
                    4  PMAP       8  DIRGRID   12  INDAT
-----------------------------------------------------------------------------




                            select column ..
```

Enter a * to select all columns in the table.

```
                                  AREA

                    1* AREAC      5  XZERO      9  DEV
                    2  AREAN      6  YZERO     10  RAKDEF
                    3  MAPNAME    7  DIRGRID   11  COMMENT
                    4  PMAP       8  DIRGRID   12  INDAT
---------------------------------------------------------------------------
AREAC AREAN          MAPNAME           PMAP     XZERO    YZERO    ZZERO
DIRGRID   DEV RAKDEF
COMMENT
INDAT
-----------------------------------------------------------------------------


                            select column ..
```

Enter <CR> to finish the selection of columns.



```
                         AREA

          1* AREAC       5  XZERO        9  DEV
          2  AREAN       6  YZERO       10  RAKDEF
          3  MAPNAME     7  DIRGRID     11  COMMENT
          4  PMAP        8  DIRGRID     12  INDAT
---------------------------------------------------------------
AREAC AREAN          MAPNAME          PMAP     XZERO    YZERO    ZZERO
DIRGRID   DEV RAKDEF
COMMENT
INDAT
---------------------------------------------------------------



                    enter select statement
where: .....................................................

---------------------------------------------------------------
```

Enter :1 = KM to select the record where AREAC is equal to the string KM.

Available operators are:

=	equal
<>	not equal
>	greater than
>=	greater than or equal
<	less than
<=	less than or equal
between .. and ..	range
not between .. and ..	range

which can be applied to both numerical and character columns and

LIKE	string matching pattern
NOT LIKE	string not matching pattern
CO	contain
BW	begin with
NB	not begin with
NC	not contain

The following operators are equal:

CO = LIKE *string*
BW = LIKE string*
NB = NOT LIKE string*
NC = NOT LIKE *string*

which are valid only for character columns only, and the logical operators

AND logical and

OR logical or

Finish with an empty line and the selected records will be displayed. You can direct the output to the screen or to a temporary table.

```
                              AREA

            1* AREAC        5  XZERO         9  DEV
            2  AREAN        6  YZERO        10  RAKDEF
            3  MAPNAME      7  DIRGRID      11  COMMENT
            4  PMAP         8  DIRGRID      12  INDAT
-----------------------------------------------------------------------------
AREAC AREAN          MAPNAME          PMAP     XZERO    YZERO    ZZERO
DIRGRID   DEV RAKDEF
COMMENT
INDAT
-----------------------------------------------------------------------------
:1 = KM



                          enter select statement
where: ...............................................................

-----------------------------------------------------------------------------
```

<CR> finish the where condition string

Next select the outputmedia.

```
output to?  (Screen, Temptable, Both, Formatted file).
```

<CR>, S or s will display the result on the terminal.

```
----------------------------------------------------------------------------
AREAC AREAN          MAPNAME          PMAP     XZERO    YZERO    ZZERO
DIRGRID   DEV RAKDEF
COMMENT
INDAT
----------------------------------------------------------------------------
KM     KAMLUNGE       25M              9E      7345320  1811340
  -6.50   1.7  F

861021
---



1 row found
ok? y
```

```
R>
```

Exercise: Select table area with less than all columns and with data from all areas, create also a temptable.

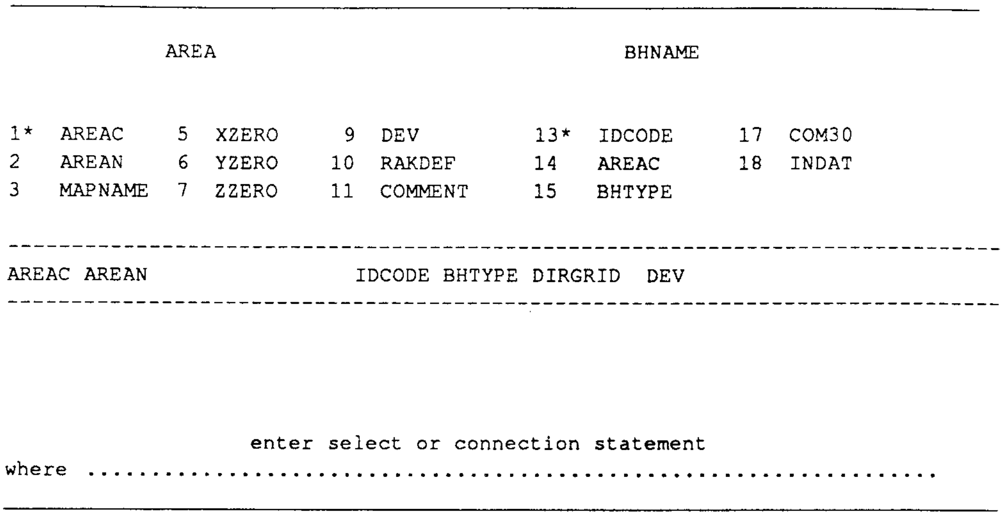
Example – Join Between Two Tables

```
R> sel area bhname;
```

```
            AREA                                    BHNAME

       AREA                                    BHNAME

1*  AREAC    5  XZERO    9  DEV          13*  IDCODE      17  COM30
2   AREAN    6  YZERO   10  RAKDEF       14   AREAC       18  INDAT
3   MAPNAME  7  ZZERO   11  COMMENT      15   BHTYPE

-----------------------------------------------------------------------------
AREAC AREAN                IDCODE BHTYPE DIRGRID  DEV
-----------------------------------------------------------------------------



                        select column ..
```

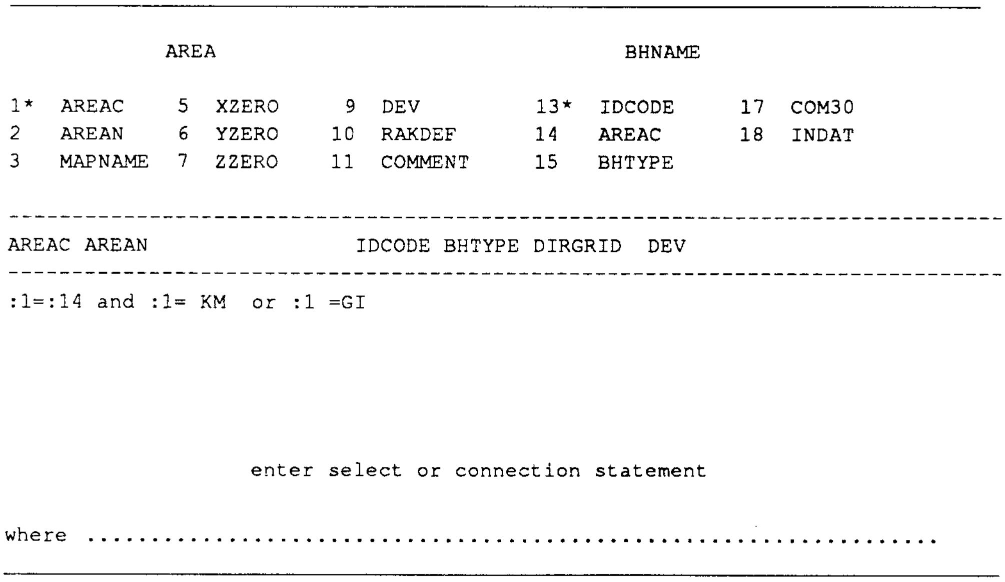
Enter column number to select column, all columns can be selected with .

Enter <CR> to finish selection of columns



```
            AREA                                    BHNAME


1*  AREAC    5  XZERO     9  DEV          13*  IDCODE      17  COM30
2   AREAN    6  YZERO    10  RAKDEF       14   AREAC       18  INDAT
3   MAPNAME  7  ZZERO    11  COMMENT      15   BHTYPE

------------------------------------------------------------------------------
AREAC AREAN                 IDCODE BHTYPE DIRGRID  DEV
------------------------------------------------------------------------------



                    enter select or connection statement
where ...........................................................
```

Enter :1=:14 and :1= KM or :1 =GI ; to connect areac in area and bhname and select area KM or GI



```
            AREA                                    BHNAME

1*  AREAC    5  XZERO     9  DEV          13*  IDCODE      17  COM30
2   AREAN    6  YZERO    10  RAKDEF       14   AREAC       18  INDAT
3   MAPNAME  7  ZZERO    11  COMMENT      15   BHTYPE

------------------------------------------------------------------------------
AREAC AREAN                 IDCODE BHTYPE DIRGRID  DEV
------------------------------------------------------------------------------
:1=:14 and :1= KM  or :1 =GI



                    enter select or connection statement

where ...........................................................
```

<CR> finish the where condition string

Next select the outputmedia.

```
output to?  (Screen, Temptable, Both, Formatted file).
```

<CR>, S or s will display the result on the terminal.

```
-----------------------------------------------------------------------------
AREAC AREAN                      IDCODE BHTYPE DIRGRID  DEV
-----------------------------------------------------------------------------
GI    GIDEÅ                      HGI01  H         -3.00 -1.2
GI    GIDEÅ                      HGI02  H         -3.00 -1.2
GI    GIDEÅ                      HGI03  H         -3.00 -1.2
GI    GIDEÅ                      HGI04  H         -3.00 -1.2
GI    GIDEÅ                      HGI05  H         -3.00 -1.2
GI    GIDEÅ                      HGI06  H         -3.00 -1.2
GI    GIDEÅ                      HGI07  H         -3.00 -1.2
GI    GIDEÅ                      HGI08  H         -3.00 -1.2
GI    GIDEÅ                      HGI09  H         -3.00 -1.2
GI    GIDEÅ                      HGI10  H         -3.00 -1.2
GI    GIDEÅ                      HGI11  H         -3.00 -1.2
GI    GIDEÅ                      HGI12  H         -3.00 -1.2
.
.
.
.
.
.
KM    KAMLUNGE                   KKM11  K         -6.50  1.7
KM    KAMLUNGE                   KKM12  K         -6.50  1.7
KM    KAMLUNGE                   KKM13  K         -6.50  1.7
KM    KAMLUNGE                   KKM14  K         -6.50  1.7
KM    KAMLUNGE                   KKM15  K         -6.50  1.7
KM    KAMLUNGE                   KKM16  K         -6.50  1.7

 76 rows found

ok?  y

R>
```

Example – Join Using More Than Two Tables

```
R> sel area bhname;
```

```
_____________________________________________________________________________

           AREA                                      BHNAME

1* AREAC      5  XZERO    9 DEV                13  IDCODE          17  COM30
2  AREAN      6  YZERO    10 RAKDEF            14  AREAC           18  INDAT
3  MAPNAME    7  ZZERO    11 COMMENT           15  BHTYPE
4             8 DIRGRID   12 INDAT             16  OIDCODE
-----------------------------------------------------------------------------
IDCODE  DIRGRID DEV

-----------------------------------------------------------------------------



where :1=:14 ;
_____________________________________________________________________________
```

output media both temptable and terminal.

```
-------------------------------------------------------------------------------
IDCODE DIRGRID  DEV
-------------------------------------------------------------------------------
HAS01   -11.80 -0.8
HAS02   -11.80 -0.8
HAS03   -11.80 -0.8
HAS04   -11.80 -0.8
HAS05   -11.80 -0.8
HAS06   -11.80 -0.8
HAS07   -11.80 -0.8
HAV01   -11.80 -0.8
HAV02   -11.80 -0.8
HAV03   -11.80 -0.8
HAV04   -11.80 -0.8
HAV05   -11.80 -0.8
HAV06   -11.80 -0.8
HAV07   -11.80 -0.8
HAV08   -11.80 -0.8
KAV01   -11.80 -0.8
KAV02   -11.80 -0.8
KAV03   -11.80 -0.8
.
.
.
.
.
.
HSV7A   -25.50 -0.3
HSV7B   -25.50 -0.3
KSV01   -25.50 -0.3
KSV02   -25.50 -0.3
KSV03   -25.50 -0.3
KSV04   -25.50 -0.3
KSV05   -25.50 -0.3
KSV07   -25.50 -0.3
KTA01
KYD04
KYD05
268 rows loaded
268 rows found
```

Next step is to join between the result of the first join and the 3:d table, here devangle.

```
R> sel abh devangle;
```

```
          ABH                        DEVANGLE

      1 IDCODE    *           4* IDCODE          8  YCOORD
      2 DIRGRID               5  BHLEN           9  DECLIN
      3 DEV                   6  VERTDEP        10 INCLIN
                              7  COORD          11 INDAT


------------------------------------------------------------------------------
IDCODE   BHLEN (m)  XCOORD YCOORD VERTDEP DECLIN INCLIN DIRGRID DEV
------------------------------------------------------------------------------



where :1=:4 ;
```

Output media both temptable and terminal.

```
----------------------------------------------------------------------
IDCODE BHLEN (m)  XCOORD  YCOORD VERTDEP (m) DECLIN INCLIN DIRGRID  DEV
----------------------------------------------------------------------
HAS01      10.00                          315.0  60.7  -11.80 -0.8
HAS01      20.00                          314.0  61.0  -11.80 -0.8
HAS01      30.00                          314.0  61.6  -11.80 -0.8
HAS01      40.00                          312.0  61.2  -11.80 -0.8
HAS01      50.00                          311.0  61.7  -11.80 -0.8
HAS01      60.00                          312.0  62.5  -11.80 -0.8
HAS01      70.00                          310.0  63.7  -11.80 -0.8
HAS01      80.00                          312.0  64.8  -11.80 -0.8
HAS01      90.00                          314.0  65.5  -11.80 -0.8
HAS01      98.00                          315.0  66.2  -11.80 -0.8
HAS01     100.00                          315.0  66.2  -11.80 -0.8
HAS02      10.00                          186.0  55.4  -11.80 -0.8
HAS02      20.00                          185.0  56.5  -11.80 -0.8
HAS02      30.00                          185.0  57.6  -11.80 -0.8
HAS02      40.00                          187.0  58.4  -11.80 -0.8
HAS02      50.00                          187.0  60.3  -11.80 -0.8
HAS02      60.00                          185.0  61.6  -11.80 -0.8
.
.
.
.
.
.
.
KTA01    600.00                           120.5  84.0
KTA01    620.00                           120.0  84.1
KTA01    630.00                           119.0  84.1
KTA01    640.00                           118.0  84.1
KTA01    650.00                           116.0  83.8
KTA01    660.00                           116.5  84.5
KTA01    670.00                           117.5  84.5
KTA01    680.00                           181.5  85.0
KTA01    690.00                           122.0  85.0
4829 rows loaded
4829 rows found
ok? y
R>
```

Example – Outer Join

This example is an outer join between table bhname and devangle. The result is a list on those borehole that are abcent in table devangle.

```
R> sel bhname devangle;

________________________________________________________________________________

                 BHNAME                                DEVANGLE

     1 IDCODE   *           5 COM30          7 IDCODE    *       11 YCOORD
     2 AREAC    *           6 INDAT          8 BHLEN     *       12 DECLIN
     3 BHTYPE                                9 VERTDEP           13 INCLIN
     4 OIDCODE                              10 XCOORD            14 INDAT
--------------------------------------------------------------------------------
IDCODE
--------------------------------------------------------------------------------




    where ~ :1=:7
________________________________________________________________________________
```

Output media both temptable and terminal.

```
--------------------------------------------------------------------------------
IDCODE
--------------------------------------------------------------------------------
BFI01
BFI02
HAV08
HFI02
HFI03
HFI04
HFI06
HFJ01
HFJ02
HFJ03
HFJ04
HFJ05
HFJ06
.
.
.
.
.
KKR03
KLA00
KST01
KST02
KSV02
KYD04
KYD05
K[V01
K[V02
137 rows loaded
137 rows found
```

List of SKB reports

Annual Reports

1977-78
TR 121
KBS Technical Reports 1 – 120
Summaries
Stockholm, May 1979

1979
TR 79-28
The KBS Annual Report 1979
KBS Technical Reports 79-01 – 79-27
Summaries
Stockholm, March 1980

1980
TR 80-26
The KBS Annual Report 1980
KBS Technical Reports 80-01 – 80-25
Summaries
Stockholm, March 1981

1981
TR 81-17
The KBS Annual Report 1981
KBS Technical Reports 81-01 – 81-16
Summaries
Stockholm, April 1982

1982
TR 82-28
The KBS Annual Report 1982
KBS Technical Reports 82-01 – 82-27
Summaries
Stockholm, July 1983

1983
TR 83-77
The KBS Annual Report 1983
KBS Technical Reports 83-01 – 83-76
Summaries
Stockholm, June 1984

1984
TR 85-01
Annual Research and Development Report 1984
Including Summaries of Technical Reports Issued during 1984. (Technical Reports 84-01 – 84-19)
Stockholm, June 1985

1985
TR 85-20
Annual Research and Development Report 1985
Including Summaries of Technical Reports Issued during 1985. (Technical Reports 85-01 – 85-19)
Stockholm, May 1986

1986
TR 86-31
SKB Annual Report 1986
Including Summaries of Technical Reports Issued during 1986
Stockholm, May 1987

1987
TR 87-33
SKB Annual Report 1987
Including Summaries of Technical Reports Issued during 1987
Stockholm, May 1988

1988
TR 88-32
SKB Annual Report 1988
Including Summaries of Technical Reports Issued during 1988
Stockholm, May 1989

1989
TR 89-40
SKB Annual Report 1989
Including Summaries of Technical Reports Issued during 1989
Stockholm, May 1990

Technical Reports
List of SKB Technical Reports 1990

TR 90-01
FARF31 –
A far field radionuclide migration code for use with the PROPER package
Sven Norman[1], Nils Kjellbert[2]
[1]Starprog AB
[2]SKB AB
January 1990

TR 90-02
Source terms, isolation and radiological consequences of carbon-14 waste in the Swedish SFR repository
Rolf Hesböl, Ignasi Puigdomenech, Sverker Evans
Studsvik Nuclear
January 1990

TR 90-03
Uncertainties in repository performance from spatial variability of hydraulic conductivities –
Statistical estimation and stochastic simulation using PROPER
Lars Lovius[1], Sven Norman[1], Nils Kjellbert[2]
[1]Starprog AB
[2]SKB AB
February 1990

TR 90-04
Examination of the surface deposit on an irradiated PWR fuel specimen subjected to corrosion in deionized water
R. S. Forsyth, U-B. Eklund, O. Mattsson, D. Schrire
Studsvik Nuclear
March 1990

TR 90-05
Potential effects of bacteria on radionuclide transport from a Swedish high level nuclear waste repository
Karsten Pedersen
University of Gothenburg, Department of General and Marine Microbiology, Gothenburg
January 1990

TR 90-06
Transport of actinides and Tc through a bentonite backfill containing small quantities of iron, copper or minerals in inert atmosphere
Yngve Albinsson, Birgit Sätmark,
Ingemar Engkvist, W. Johansson
Department of Nuclear Chemistry,
Chalmers University of Technology, Gothenburg
April 1990

TR 90-07
Examination of reaction products on the surface of UO_2 fuel exposed to reactor coolant water during power operation
R. S. Forsyth, T. J. Jonsson, O. Mattsson
Studsvik Nuclear
March 1990

TR 90-08
Radiolytically induced oxidative dissolution of spent nuclear fuel
Lars Werme[1], Patrik Sellin[1], Roy Forsyth[2]
[1]Swedish Nuclear Fuel and waste Management Co (SKB)
[2]Studsvik Nuclear
May 1990

TR 90-09
Individual radiation doses from unit releases of long lived radionuclides
Ulla Bergström, Sture Nordlinder
Studsvik Nuclear
April 1990

TR 90-10
Outline of regional geology, mineralogy and geochemistry, Poços de Caldas, Minas Gerais, Brazil
H. D. Schorscher[1], M. E. Shea[2]
[1]University of Sao Paulo
[2]Battelle, Chicago
December 1990

TR 90-11
Mineralogy, petrology and geochemistry of the Poços de Caldas analogue study sites, Minas Gerais, Brazil
I: Osamu Utsumi uranium mine
N. Waber[1], H. D. Schorscher[2], A. B. MacKenzie[3], T. Peters[1]
[1]University of Bern
[2]University of Sao Paulo
[3]Scottish Universities Research & Reactor Centre (SURRC), Glasgow
December 1990

TR 90-12
Mineralogy, petrology and geochemistry of the Poços de Caldas analogue study sites, Minas Gerais, Brazil
II: Morro do Ferro
N. Waber
University of Bern
December 1990

TR 90-13
Isotopic geochemical characterisation of selected nepheline syenites and phonolites from the Poços de Caldas alkaline complex, Minas Gerais, Brazil
M. E. Shea
Battelle, Chicago
December 1990

TR 90-14
Geomorphological and hydrogeological features of the Poços de Caldas caldera, and the Osamu Utsumi mine and Morro do Ferro analogue study sites, Brazil
D. C. Holmes[1], A. E. Pitty[2], R. Noy[1]
[1]British Geological Survey, Keyworth
[2]INTERRA/ECL, Leicestershire, UK
December 1990

TR 90-15
Chemical and isotopic composition of groundwaters and their seasonal variability at the Osamu Utsumi and Morro do Ferro analogue study sites, Poços de Caldas, Brazil
D. K. Nordstrom[1], J. A. T. Smellie[2], M. Wolf[3]
[1]US Geological Survey, Menlo Park
[2]Conterra AB, Uppsala
[3]Gesellschaft für Strahlen- und Umweltforschung (GSF), Munich
December 1990

TR 90-16
Natural radionuclide and stable element studies of rock samples from the Osamu Utsumi mine and Morro do Ferro analogue study sites, Poços de Caldas, Brazil
A. B. MacKenzie[1], P. Linsalata[2], N. Miekeley[3],
J. K. Osmond[4], D. B. Curtis[5]
[1]Scottish Universities Research & Reactor Centre (SURRC), Glasgow
[2]New York Medical Centre
[3]Catholic University of Rio de Janeiro (PUC)
[4]Florida State University
[5]Los Alamos National Laboratory
December 1990

TR 90-17
Natural series nuclide and rare earth element geochemistry of waters from the Osamu Utsumi mine and Morro do Ferro analogue study sites, Poços de Caldas, Brazil
N. Miekeley[1], O. Coutinho de Jesus[1],
C-L Porto da Silveira[1], P. Linsalata[2], J. N. Andrews[3],
J. K. Osmond[4]
[1]Catholic University of Rio de Janeiro (PUC)
[2]New York Medical Centre
[3]University of Bath
[4]Florida State University
December 1990

TR 90-18
Chemical and physical characterisation of suspended particles and colloids in waters from the Osamu Utsumi mine and Morro do Ferro analogue study sites, Poços de Caldas, Brazil
N. Miekeley[1], O. Coutinho de Jesus[1],
C-L Porto da Silveira[1], C. Degueldre[2]
[1]Catholic University of Rio de Janeiro (PUC)
[2]PSI, Villingen, Switzerland
December 1990

TR 90-19
Microbiological analysis at the Osamu Utsumi mine and Morro do Ferro analogue study sites, Poços de Caldas, Brazil
J. West[1], A. Vialta[2], I. G. McKinley[3]
[1]British Geological Survey, Keyworth
[2]Uranio do Brasil, Poços de Caldas
[3]NAGRA, Baden, Switzerland
December 1990

TR 90-20
Testing of geochemical models in the Poços de Caldas analogue study
J. Bruno[1], J. E. Cross[2], J. Eikenberg[3], I. G. McKinley[4],
D. Read[5], A. Sandino[1], P. Sellin[6]
[1]Royal Institute of Technology (KTH), Stockholm
[2]AERE, Harwell, UK
[3]PSI, Villingen, Switzerland
[4]NAGRA, Baden, Switzerland
[5]Atkins, ES, Epsom, UK
[6]Swedish Nuclear and Waste Management Co (SKB), Stockholm
December 1990

TR 90-21
Testing models of redox front migration and geochemistry at the Osamu Utsumi mine and Morro do Ferro analogue sites, Poços de Caldas, Brazil
J. Cross[1], A. Haworth[1], P. C. Lichtner[2],
A. B. MacKenzi[3], L. Moreno[4], I. Neretnieks[4],
D. K. Nordstrom[5], D. Read[6], L. Romero[4],
S. M. Sharland[1], C. J. Tweed[1]
[1]AERE, Harwell, UK
[2]University of Bern
[3]Scottish Universities Research & Reactor Centre (SURRC), Glasgow
[4]Royal Institute of Technology (KTH), Stockholm
[5]US Geological Survey, Menlo Park
[6]Atkins ES, Epsom, UK
December 1990

TR 90-22
Near-field high temperature transport: Evidence from the genesis of the Osamu Utsumi uranium mine analogue site, Poços de Caldas, Brazil
L. M. Cathles[1], M. E. Shea[2]
[1]University of Cornell, New York
[2]Battelle, Chicago
December 1990

TR 90-23
Geochemical modelling of water-rock interactions at the Osamu Utsumi mine and Morro do Ferro analogue sites, Poços de Caldas, Brazil
D. K. Nordstrom[1], I. Puigdomenech[2], R. H. McNutt[3]
[1]US Geological Survey, Menlo Park
[2]Studsvik Nuclear, Sweden
[3]McMaster University, Ontario, Canada
December 1990

TR 90-24
The Poços de Caldas Project: Summary and implications for radioactive waste management
N. A. Chapman[1], I. G. McKinley[2], M. E. Shea[3],
J. A. T. Smellie[4]
[1] INTERRA/ECL, Leicestershire, UK
[2] NAGRA, Baden, Switzerland
[3] Battelle, Chicago
[4] Conterra AB, Uppsala

TR 90-25
Kinetics of UO_2(s) dissolution reducing conditions:
numerical modelling
I. Puigdomenech[1], I. Casas[2], J. Bruno[3]
[1] Studsvik AB, Nyköping, Sweden
[2] Department of Chemical Engineering, E.T.S.E.I.B. (U.P.C.), Barcelona, Spain
[3] Department of Inorganic Chemistry, The Royal Institute of Technology, Stockholm, Sweden
May 1990

TR 90-26
The effect from the number of cells, pH and lanthanide concentration on the sorption of promethium on gramnegative bacterium (Shewanella Putrefaciens)
Karsten Pedersen[1], Yngve Albinsson[2]
[1] University of Göteborg, Department of General and Marine Microbiology, Gothenburg, Sweden
[2] Chalmers University of Technology, Department of Nuclear Chemistry, Gothenburg, Sweden
June 1990

TR 90-27
Isolation and characterization of humics from natural waters
B. Allard[1], I. Arsenie[1], H. Borén[1], J. Ephraim[1],
G. Gårdhammar[2], C. Pettersson[1]
[1] Department of Water and Environmental Studies, Linköping University, Linköping, Sweden
[2] Department of Chemistry, Linköping University, Linköping, Sweden
May 1990

TR 90-28
Complex forming properties of natural organic acids.
Part 2. Complexes with iron and calcium
James H. Ephraim[1], Andrew S. Mathuthu[2],
Jacob A. Marinsky[3]
[1] Department of Water in Environment and Society, Linköping University, Linköping, Sweden
[2] Chemistry department, University of Zimbabwe, Harare, Zimbabwe
[3] Chemistry Department, State University of New York at Buffalo, Buffalo, NY, USA
July 1990

TR 90-29
Characterization of humic substances from deep groundwaters in granitic bedrock in Sweden
C. Pettersson, J. Ephraim, B. Allard, H. Borén
Department of Water and Environmental Studies, Linköping University, Linköping, Sweden
June 1990

TR 90-30
The earthquakes of the Baltic shield
Ragnar Slunga
Swedish National Defence Research Institute
June 1990

TR 90-31
Near-field performance of the advanced cold process canister
Lars Werme
Swedish Nuclear Fuel and Waste Management Co (SKB)
September 1990

TR 90-32
Radioclide transport paths in the nearfield – a KBS-3 concept study
Roland Pusch
Clay Technology AB and Lund University of Technology
July 1990

TR 90-33
PLAN 90
Costs for management of the radioactive waste from nuclear power production
Swedish Nuclear Fuel and Waste Management Co (SKB)
June 1990