

Using Editors

This section describes the Super Natural editors.

This section covers the following topics:

- What are the Super Natural Editors?
 - Editor Commands
 - Editor Help System
 - Operators Available in the Full-Screen Editors
 - The Selection Editor
 - The Calculation Editor
 - The Logical Conditions Editor
 - The SQL SELECT Editor
 - The PC File Description Editor
 - The Work File Description Editor
 - The User File Description Editor
 - Arrays
-

What are the Super Natural Editors?

An editor is a tool for creating, modifying and deleting components necessary for creating transactions. Super Natural offers full-screen editors and file description editors. The editors which you can use for a transaction depend on your transaction modes.

You can also use the Natural Report ManagerLayout Editor to edit the layout of reports during creation and modification. The Layout Editor is not related to the other Super Natural editors and is described in the sectionNatural Report Manager .

The Super Natural full-screen editors and file description editors function in the same way.

Full-Screen Editors

You use full-screen editors to tell Super Naturalwhich data you want from the database, to specify calculations which take place when the transaction is run and for logical condition processing. The following full-screen editors are available:

- Selection Editor
- SQL SELECT Editor
- Calculation Editor
- Logical Conditions Editor

Full-screen editors are empty when you invoke them for the first time.

File Description Editors

You use the file description editors to describe file layouts. The following file description editors are available:

- PC File Description Editor
- Work File Description Editor
- User File Description Editor

File description editors consist of columns.

Editor Commands

This section describes the commands available in the editors and the editors' online help system.

Line Commands

Line commands influence the lines in the editors and are structured as follows:

```
escape-character line-command-character
```

Line commands are entered at the beginning of an editor line.

Escape Character

The first character of each line command is the escape character defined for you. The default escape character is a period (.). The default escape character is overridden if a different entry is made in an existing Natural editor profile for you or for Super Natural.

To check or modify the escape character in the Natural editor profile, enter PROFILE in the Natural editor command line. Then select Additional Options and Editor Defaults. Here you can check or modify the Escape Character for Line Command.

Line-Command Character

You can use some line-command characters in combination with a repetition factor or with a marker.

(Blank)

Entering a blank at the end of each line command prevents the editor from trying to interpret any of the information in the editor line as part of the line command.

Issuing Line Commands

To issue a line command

1. Type the escape character at the first position of an editor line.
2. Type the line command character immediately after the escape character.
3. Type a blank.
4. Press Enter.

Do not leave a blank between the escape character and the line command character.

List of Line Commands

The line commands available in the Super Natural editors are listed below:

Note:

The notation "n" indicates a repetition factor. The default repetition factor is 1. You must enclose the repetition factor in parentheses.

Note:

The line in which you enter the command is referred to as the current line.

Command Function

<code>.C</code>	Copies the current line and inserts it directly below the current line.
<code>.C(n)</code>	Copies the current line n times and inserts the new lines directly below the current line.
<code>.CX</code>	Copies the line previously marked with X and inserts it directly below the current line. For further information on marking lines, see the description of the <code>.X</code> command.
<code>.CX(n)</code>	Copies the line previously marked with X n times and inserts the new lines directly below the current line. For further information on marking lines, see the description of the <code>.X</code> command.
<code>.CY</code>	Copies the line previously marked with Y and inserts it directly below the current line. For further information on marking lines, see the description of the <code>.Y</code> command.
<code>.CY(n)</code>	Copies the line previously marked with Y n times and inserts the new lines directly below the current line. For further information on marking lines, see the description of the <code>.Y</code> command.
<code>.CX-Y</code>	Copies the block of lines previously delimited by the X and Y markers and inserts it directly below the current line. For further information on marking lines, see the descriptions of the <code>.X</code> and the <code>.Y</code> commands.
<code>.CX-Y(n)</code>	Copies the block of lines previously delimited by the X and Y markers n times and inserts it directly below the current line. For further information on marking lines, see the descriptions of the <code>.X</code> and the <code>.Y</code> commands.
<code>.D</code>	Deletes the current line.
<code>.D(n)</code>	Deletes n lines starting with the current line.
<code>.I</code>	Inserts a default of 6 blank lines directly below the current line. Lines that are left blank are eliminated when you next press <code>.</code>
<code>.I(n)</code>	Inserts n empty lines after the current line, where n can be in the range from 1 to 13. Lines that are left blank are eliminated when you next press <code>.</code>
<code>.L</code>	Undoes all modifications you have made to the line since you last pressed <code>.</code>
<code>.MX</code>	Moves the line previously marked with X and inserts it directly below the current line. For further information on marking lines, see the description of the <code>.X</code> command.
<code>.MY</code>	Moves the line previously marked with Y and inserts it directly below the current line. For further information on marking lines, see the description of the <code>.Y</code> command.
<code>.MX-Y</code>	Moves the block of lines previously delimited by the X and Y markers and inserts it directly below the current line. For further information on marking lines, see the descriptions of the <code>.X</code> and the <code>.Y</code> commands. When you move lines using the <code>.M</code> commands, the lines below automatically move up to fill the gap which would otherwise be left.
<code>.P</code>	Scrolls so that the current line appears at the top of the screen.
<code>.S</code>	Splits the current line at the cursor position. The content of the current line after the cursor position is written into the beginning of a new line which is created below the current line. Not allowed in the Logical Conditions Editor.
<code>.X</code>	Marks a line or delimits the beginning of a block of lines.
<code>.Y</code>	Marks a line or delimits the end of a block of lines. A line marked with both X and Y using the <code>.X</code> and <code>.Y</code> commands, is indicated by Z. If you want to use the line with the <code>.C</code> or the <code>.M</code> command, you must still reference it with either X or Y.

Editor Direct Commands

Editor direct commands are structured as follows:

```
EDIT editor-direct-command
```

The key word EDIT must precede every editor direct command.

Issue an Editor Direct Command

▶ **To issue an editor direct command:**

- Enter the command in the command line of any Super Natural editor except for the Layout Editor.

List of Editor Direct Commands

Command	Description
ADD	Adds 6 empty lines at the end of the lines.
ADD (n)	Adds n empty lines at the end of the lines. The value n must be in the range from 1 to 6.
BOTTOM	Positions to the end of the lines.
CHANGE 'value1'value2'	Scans for the string entered as value 1 and replaces it with the string entered as value 2. Values which have been replaced are marked with R. Not allowed in the Logical Conditions editor.
CLEAR	Deletes all the editor lines.
DX	Deletes the line marked with X.
DY	Deletes the line marked with Y.
DX-Y	Deletes the block of lines delimited by X and Y.
EX	Deletes all the lines above the line marked with X.
EY	Deletes all the lines below the line marked with Y.
EX-Y	Deletes all the lines except for the block of lines delimited by X and Y.
LET	Undoes all modifications made since you last pressed .
RESET	Deletes all line markers except for S and R.
SCAN value'	Scrolls to the string entered as value'. The single quotation marks are optional. Lines containing strings found by the SCAN value' command are marked with S.
SCAN =	Scrolls to the next occurrence of the string entered as value'.
TOP	Positions to the top of the lines in the editor.
X	Scrolls to the line marked with X.
Y	Scrolls to the line marked with Y.
+H	Pages half a page forwards.
-H	Pages half a page backwards.
+n	Positions n lines forwards.
-n	Positions n lines backwards.
+P	Pages one page forward.
-P	Pages one page backwards.

Note:

You can also use the Super Natural Paging functions for positioning within the editors.

The S Marker and the R Marker

The S marker and the R marker overwrite any other markers previously set. Lines are marked with S or R until the SCAN command or CHANGE 'value1'value2' command respectively is deactivated by issuing another command. The original markers become visible again.

General Editor Commands

The commands listed below are only available in the Super Natural editors.

Command	Description
CHECK	Checks your entries in all the Super Natural editors apart from the User File Description Editor and pin-points errors. You can correct the error, ignore it and continue working using CANCEL, invoke syntax help by pressing or leave the editor and save the error using EXIT.
CONTINUE	Invokes a syntax help window which tells you what you can do next. Available in all the full-screen editors apart from the SQL SELECT Editor. If you issue the CONTINUE command when there is an error in the editor lines, the error is pin-pointed and you can either correct it or invoke syntax help as for the CHECK command. The description of Sample Transaction 1 in the Transaction in the Tutorial demonstrates how to use the Continue function. For further information on using the CONTINUE command in the Logical Conditions Editor, see The Logical Conditions Editor in Using Editors .
DELETE	Deletes all the editor lines. This achieves the same as the EDIT CLEAR command.
FIELDS	Invokes the Field Selection List window. The fields which are already in use are listed first and marked with a chevron (>). You can page forward to list the other fields in the file(s) you are using.
REFRESH	Deletes the lines in the current editor and recalls the lines which were present (if any) when the transaction was last saved.
TABLES	Invokes the List DB2 Tables window. Issue from the SQL SELECT editor.

Editor Help System

Two types of help are available in the Super Natural editors:

Help for Editor Commands

To invoke the help for editor commands:

1. Position the cursor in the editor area.
2. Press PF1.

A help screen appears where you can choose between displaying information on editor line commands and editor direct commands.

Syntax Help

Syntax help is available in all the full-screen editors apart from the SQL SELECT Editor. Syntax help does not just display information, but allows you to make entries which are then transferred to the underlying screen.

You can invoke syntax help if you don't know what to do next.

To invoke syntax help if you don't know what to do next:

- Issue the CONTINUE command.

You can also invoke syntax help if the CHECK command has pin-pointed an error which you do not know how to correct.

 **To invoke syntax help if you have just issued the CHECK command:**

- Press Enter.

In both cases, a window appears telling you which alternatives are available. Syntax help windows display selection lists of objects or options. Selecting an object or option may lead to a further syntax help window, may invoke another Super Natural window eg. the Define User Field window or may result in an entry being inserted in the editor lines.

Many syntax help windows display the option Mark to Replace. If you mark this option, the entry you have selected will replace the error in the editor lines instead of being inserted.

Note:

The help system offered in the SQL SELECT Editor is different to the help offered in the other Super Natural editors and is described later in this section.

Operators Available in the Full-Screen Editors

The following operators are available in the full-screen editors:

	Selection Statement	Calculation Statement	Logical	Condition	Statement
Operator			IF Clause	THEN Clause	ELSE Clause
EQ (=)	X	X	X	X	X
NE (^=)	X		X		
LT (<)	X		X		
LE (<=)	X		X		
GT (>)	X		X		
GE (>=)	X		X		
THRU	X		X		
ST	X				
NOT	X				
AND	X		X		
OR	X		X		
OR=	X		X		
+		X		X	X
-		X		X	X
/		X		X	X
*		X		X	X
		X		X	X

Note:

If your keyboard does not have the vertical bar character, your system administrator will tell you which character to use.

The Selection Editor

You use the Selection Editor for entering selection criteria in reporting transactions with the transaction mode Data Selection = FULL-SCREEN. Selection criteria tell Super Natural which data you want from the database.

▶ To invoke the Selection Editor:

- Issue the SELECTION command.

The SELECTION command is assigned to PF10.

The Selection Editor appears:

```

15:27                      ***** Super Natural *****                      05.Jan.1993
SNZUL-S                      - Selection Editor -                               Tuesday

      1 <                                                            > 1
      2 <                                                            > 2
      3 <                                                            > 3
      4 <                                                            > 4
      5 <                                                            > 5
      6 <                                                            > 6
      7 <                                                            > 7
      8 <                                                            > 8
      9 <                                                            > 9
     10 <                                                            > 10
     11 <                                                            > 11
     12 <                                                            > 12
     13 <                                                            > 13
     14 <                                                            > 14
     15 <                                                            > 15

Define selection lines or use CONTINUE
Command ===>
Enter-PF13--PF14--PF15--PF16--PF17--PF18--PF19--PF20--PF21--PF22--PF23--PF24---
      Lay  Chlst Menu  Opt  Modes      --  ++

```

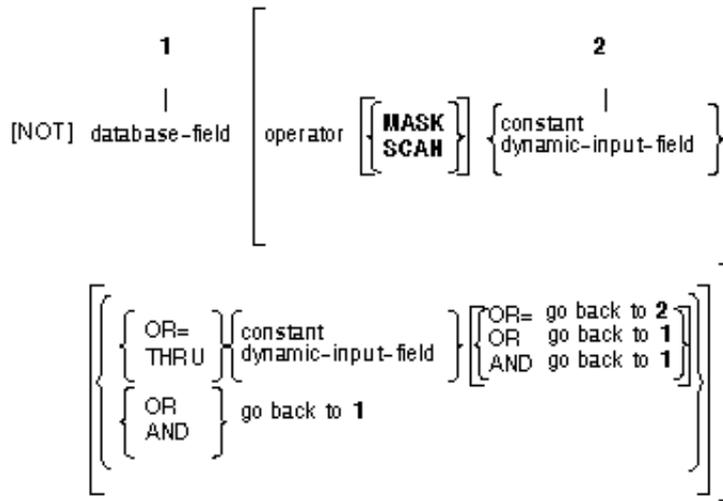
The following sections describe how to construct selection criteria. The examples are taken from the demonstration file SAG-TOURS-E-YACHT.

Selection criteria consist of one or more selection statements which are structured as follows:

$$[\text{NOT}] \text{ database-field } \left[\text{operator } \left\{ \begin{array}{l} \text{value} \\ \text{function value} \end{array} \right\} \right]$$

You can combine selection statements using the logical operators AND and OR.

The following is a more detailed syntax diagram describing selection statements:

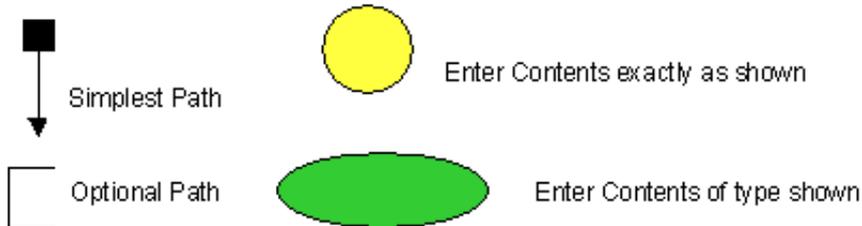


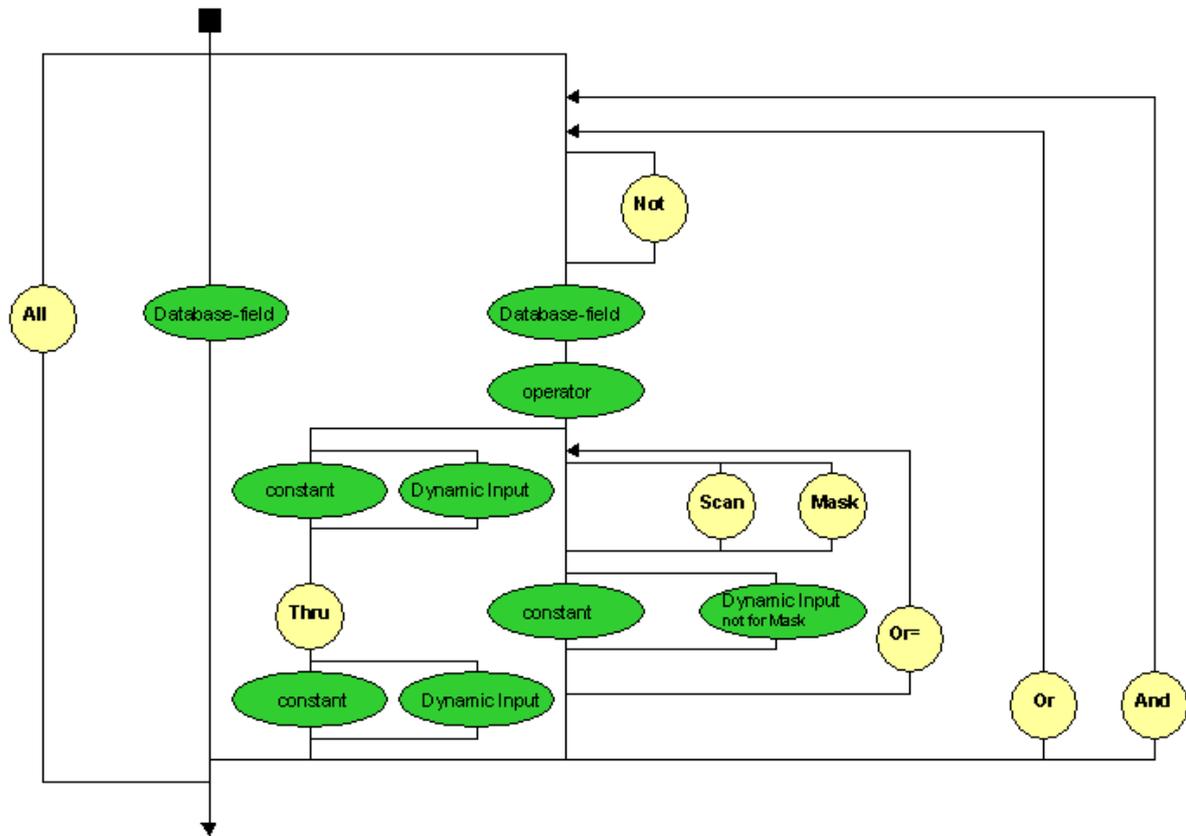
This documentation uses the Natural syntax conventions which are described in the Preface.

Note:

The CHECK and theCONTINUE functions tell you exactly what you can enter in the situation you are in.

The following is the key for the alternative syntax diagram:





Database Field

You can either use the full field name or the field reference shown in the REF column of the Worksheet.

If a field name occurs in two or more of the files you are using, you must always use the field reference.

You may not use user fields, groups and periodic groups here. You can, however, use a field contained within a group.

The performance of Super Natural is improved if you use key fields as selection criteria.

Operators

The following operators are available:

Operator	Meaning	Records Returned
EQ or =	Equals	Where the specified field has the specified value
NE or ^=	Not equal to	Where the specified field does not have the specified value
GT or >	Greater than	Where the value of the specified field is greater than the specified value
LT or <	Less than	Where the value of the specified field is less than the specified value
GE or >=	Greater than or equal to	Where the value of the specified field is greater than or equal to the specified value
LE or <=	Less than or equal to	Where the value of the specified field is less than or equal to the specified value
ST	Starting with	Where the value of the specified field starts with the specified string

Extended Operators

The following operators are available:

Operator	Meaning	Records Returned
THRU	Range of values	Where the specified field has the specified range of values
OR=	Value enumeration	Where the specified field has all specified values

THRU

You use the extended operator THRU to search for a range of values.

Example:

```
YACHT-NAME = 'A' THRU 'F'
```

This returns all the records for yachts whose names start with A, B, C, D or E, and any yacht with the name F.

Note:

Values starting with FA are not returned.

OR=

To search for more than one value for a field, use the extended operator *OR =* between each of the values.

Example:

```
YACHT-TYPE = 'PACIFIC 29' OR= 'PACIFIC 315' OR=
'PACIFIC 325' AND ID-CH-BASE >'11'
```

This finds records containing yachts of type Pacific 29, Pacific 315 and Pacific 325 and from that group returns those whose charter base ID is the number twelve or above.

Value

After the operator you can enter either a constant or a dynamic input field depending on the context. You can either enter the value directly or issue the CONTINUE command if you are not sure which kind of value to use.

▶ **To find out which types of values are relevant to your present situation:**

- Issue the CONTINUE command.

Constants

For further information on database fields and constants, see **Defining Fields**.

Dynamic Input Fields

You can use dynamic input fields if you want to execute the transaction several times using different values for a field or fields each time.

▶ **To create a dynamic input field:**

- Enter the hash sign (#) where you would normally enter the value.

Example:

```
YACHT-NAME = #
```

The Define User Field window appears with a hash sign (#) in the Dynamic Input field when you issue either the CHECK or CONTINUE command:

```

15:29          ***** Super Natural *****          05.Jan.1993
SNZUL-S          - Selection Editor -          Tuesday

          +-----Define User Field-----+
1 < YACHT-NAME = # ! Field:          !
2 <          !          !          !
3 <          ! Name or value .. _____ !
4 <          ! Format ..... A          !
5 <          ! Length ..... 30_       !
6 <          ! Dec. places .... _      !
7 <          ! Subfield of .... _____ !
8 <          ! Offset ..... _____ !
9 <          ! Dynamic input .. #      !
10 <         !          !          !
11 <         +-----+
12 <                                     > 12

```

Make your entries in the Define User Field window as described in the section **Fields**.

You have now defined a dynamic input field.

Note:

When you return to the Selection Editor, the hash (#) sign is replaced by the user field name you have just defined.

When you run the transaction, the Dynamic Input Value window appears as in the following example:

15:30		***** Super Natural *****	05.Jan.1993
SN3011		- Worksheet -	Tuesday
Ref DB	Field Name	+----- Dynamic Input Value -----+	
AA 1K	YACHT-ID	! Transaction: IDDM6	!
AB 1K	YACHT-NAME	!	!
AC 1	YACHT-BRANCH	! DYNAMIC .. _____	!
AD 1K	ID-CH-BASE	+-----	+-----
AE 1K	ID-S-OWNER	_____	_____

Enter the value you want your dynamic input field to have when the transaction is run this time.

The transaction is run using the value specified.

You can also use multiple dynamic input fields as shown in the examples below:

Examples:

```
WIDTH = # THRU #
LENGTH = 10 THRU #
YACHT-TYPE = # AND LENGTH GT #
YACHT-NAME = SCAN #
```

Each time you execute or run the transaction, Super Natural asks you to supply a value for each dynamic input field.

You may have problems if you run a transaction containing dynamic input fields in batch mode.

Note:

You can also use dynamic input fields in the Calculation and Logical Conditions Editors.

Functions

You can use the Scan function or theMask function in the Selection Editor after the EQUALS operator.

The Scan Function

You can use the Scan function to search for a specific string of characters located anywhere within an alphanumeric field. Use the Scan function as follows:

```
DATABASE FIELD = SCAN 'VALUE'
```

Example:

```
YACHT-NAME = SCAN 'Q'
```

This returns all the records where the value for the fieldYACHT-NAME contains the letter Q.

Mask Function

You can use the Mask function to search for a specific string of characters within specific positions of an alphanumeric field. Use the mask function as follows:

```
DATABASE FIELD = MASK 'definition'
```

The mask definition defines the position of the character string or strings you want to look for. Mark the positions whose content you want to be ignored with a period (.) and enter the values you are looking for in the positions in which you are looking for them. You do not need to enter periods after the last value.

Example:

```
YACHT-NAME = MASK '.HA...A'
```

This returns all the records where the value for the field YACHT-NAME contains the letter H in the second position and A in the third and seventh positions.

Logical Operators

The following logical operators are available and are executed in the following order:

NOT

AND

OR

Selection criteria enclosed within parentheses are evaluated first.

Example:

```
((YACHT-TYPE = 'PACIFIC 34' OR MOTOR > 60) AND ID-CH-BASE = '12' AND NOT MOTOR =55)
```

First this finds the records which have either yacht type Pacific 34 or a motor larger than 60 and from that group returns those records where the charter base ID is 12, but not one with the motor of 55.

NOT

Example:

```
YACHT-TYPE = 'ATLANTIC 32' AND NOT ID-CH-BASE = '12'
```

This returns all the records for Atlantic 32 yachts who have not the charter base ID number 12.

AND

You can combine selection criteria statements with the logical operator AND to restrict the range of records returned.

Example:

```
YACHT-TYPE = 'ATLANTIC 32' AND ID-CH-BASE >= '12'
```

This returns all the records for Atlantic 32 yachts whose charter base ID is the number twelve or above.

Note:

The field ID-CH-BASE is an alphanumeric field, so the value 12 must be enclosed within quotation marks.

OR

If you combine selection criteria statements with the logical operator OR, each statement is handled independently.

Example:

```
YACHT-TYPE = 'ATLANTIC 32' OR ID-CH-BASE >= '12'
```

This returns all the records for Atlantic 32 yachts and for yachts whose charter base ID is the number twelve or above.

Remember that OR is processed after THRU, OR= and AND.

Example :

```
YACHT-TYPE = 'PACIFIC 29' OR YACHT-TYPE = 'PACIFIC 315' OR  
YACHT-TYPE = 'PACIFIC 325' AND ID-CH-BASE >'11'
```

This first finds and returns the records of yachts of type Pacific 325 which have the charter base ID 12 or above. In addition, the records of yachts of type Pacific 29 and Pacific 315 are returned.

Compare this result with that of the example given for the OR= operator.

Note:

You can not use the OR= operator as a shorter way of entering multiple OR statements - the results are different!

Special Cases

Multiple File Considerations

You can only connect selection statements using fields from different files using the AND operator.

For example, if you are using the files SAG-Tours-E-Yacht and SAG-Tours-E-Prices, the following selection criteria statement is not allowed:

```
YACHT-TYPE = 'INDIC-CAT 32' OR PRICE-1W < 2000
```

You would have to use the following:

```
CH-YACHT-TYPE = 'INDIC-CAT 32' OR PRICE-1W < 2000
```

where the field CH-YACHT-TYPE is also from the SAG-Tours-E-Prices file.

Field Name

If you specify a field name as your only selection criteria, Super Natural reads all the records from the file which contain a value for the field specified and returns them in ascending order of value. You can only use descriptors in this manner.

Example

```
YACHT-NAME
```

This returns all the records in the file sorted according to the alphabetical order of the yacht names if no other sort criteria are specified.

You can use a logical operator to connect the selection statement *field name* to further selection statements.

You can only use the selection statement *field name* as the first selection statement.

File Linking Considerations

If you specify a field name from your primary file as your only selection criteria, Super Natural reads all the records from the primary file which contain a value for the field specified and returns them in ascending order of value.

If you specify a field name from the secondary file as your only selection criteria, Super Natural reads all the records from the primary file (READ PHYSICAL).

All

If you enter *ALL* as your only selection criteria, Super Natural returns all the records in the file or files in physical sequence (the order in which they are stored in the database) if no other sort criteria are specified.

You can not connect the selection statement *ALL* to further selection statements.

Using Phonetic Descriptors as Selection Criteria

You can use phonetic descriptors to perform phonetic searches on fields. Phonetic searches result in the return of all values which sound similar to the search value. Phonetic searches are useful if you are not sure how to spell what you are looking for!

Example

```
YACHT-NAME-PH = PERSEFFUNNY'
```

This returns all the records where the value for the field YACHT-NAME sounds like Perseffunny'.

Note:

You can not display phonetic descriptors themselves in a report.

Locking Selection Criteria

You can lock the first *n* selection lines in the Selection Editor so that they can not be modified. Only the creator of a transaction can lock selection lines.

You can add further selection criteria but you may only use the logical operator AND to connect new selection statements to the locked lines.

You can also lock the whole Selection Editor. This feature is useful if you are going to copy a transaction into a common library and you don't want other users to change the selection criteria.

You can not lock selection lines if the transaction uses a superfile.

You can not unlock locked selection lines.

To lock selection criteria:

1. Issue the LOCK command from within the Selection Editor.

You can only issue the LOCK command if the selection criteria are correct.

The Lock Selection Lines window appears:

```

15:35          ***** Super Natural *****          05.Jan.1993
SNZUL-S          - Selection Editor -                      Tuesday

                +-----Lock Selection Lines-----+
  1 < YACHT-NAME = DYN !                                     !
  2 <                                     ! Enter the number of lines which shall be locked !
  3 <                                     ! into the entry 'locked selection lines'.         !
  4 <                                     ! The value must be greater than the old value     !
  5 <                                     ! but smaller than the number of existing         !
  6 <                                     ! selection lines.                               !
  7 <                                     ! To lock the whole selection editor, enter 99.     !
  8 <                                     !                                     !
  9 <                                     ! Existing selection lines .. 1                 !
 10 <                                     ! Locked selection lines .... 0_                !
 11 <                                     !                                     !
 12 <                                     +-----+
 13 <                                     > 13

```

2. Enter the number of selection lines you want to be locked.

For example, if you enter 5, the first five lines are locked and if you enter 2, the first two are locked.

▶ To lock the whole Selection Editor:

- Enter 99.

A confirmation window appears.

▶ To confirm the lock:

- Enter "Y" in the confirmation window.

The lock comes into effect immediately, but if you leave the Selection Editor using the CANCEL command or the MENU command, the lock is lost. This only applies to the first time you leave the Selection Editor after defining a lock

Locked selection lines are presented in the same way as normal protected fields.

The Calculation Editor

You use the Calculation Editor to enter calculation statements. You can define multiple calculation statements separated by blanks.

This section covers the following topics:

- Invoking the Calculation Editor
- Calculation statement elements
- Types of calculation statement (arithmetic, assigning values to a field and joining fields)



Using Date and Time format.

Invoking the Calculation Editor



To invoke the Calculation Editor:

- Issue the CALCULATION command.

The Calculation Editor appears:

```

15:36                      ***** Super Natural *****          05.Jan.1993
SNZUL-C                    - Calculation Editor -                  Tuesday

      1 <                                                           > 1
      2 <                                                           > 2
      3 <                                                           > 3
      4 <                                                           > 4
      5 <                                                           > 5
      6 <                                                           > 6
      7 <                                                           > 7
      8 <                                                           > 8
      9 <                                                           > 9
     10 <                                                           > 10
     11 <                                                           > 11
     12 <                                                           > 12
     13 <                                                           > 13
     14 <                                                           > 14
     15 <                                                           > 15

SELECTION successfully terminated
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Next  Exit  Field Check Flip  -      +      Cont  Run  Canc

```

Calculation Statement Elements

Calculation statements consist of various mandatory and optional elements in the positions shown in the following diagram:

```

[ROUNDED]result-field= [mathematical-function] operand[operatoroperand]...

```

This documentation uses the Natural syntax conventions which are described in the Preface.

Note:

The Continue function tells you exactly what you can enter in the situation you are in.

Result Field (Mandatory)

The result field contains the result of your calculation. You can either define a user field or use a database field. For further information on user fields, see The Field Name Column(Defining User Fields and Dynamic Input Fields) in **Working with the Worksheet**.

In reporting transactions, the result of an arithmetic calculation is used for output purposes only and has no effect on the value of the field in the database. However, you can use the same type of calculation in data maintenance transactions to update data (for further information, see Working with Data Maintenance Transactions).

Equals Operator (=) (Mandatory)

The result of each statement must be introduced by the equals operator (=).

Operand (Mandatory)

An operand can either be a field or a constant according to the type of statement you are creating. For further information on fields and constants, see Defining Fields. The different types of calculation statement are described later in this section. For further information on using dynamic input fields, see Operators in Selection Editor.

ROUNDED (Optional)

You use the ROUNDED keyword as follows:

```
ROUNDED result-field
```

The result field is rounded to the field length and decimal places defined as shown below:

Value	Result Field Definition		Value in Result Field	
	Length	Decimal Places	WITHOUT Rounding	WITH Rounding
3.87654321	5	3	3.876	3.877
4.49	5	3	4.49	4.49
4.49	5	0	4	5

Example:

```
ROUNDED RESULT = PERCENT (BUNKS)
```

RESULT is a numeric user field with Field Length 3 and Dec. Places 2. If you run a transaction using YACHT-NAME ST P' as your selection criteria, the field RESULT shows how many percent of the total number of bunks available each yacht whose name starts with P contains with the number rounded to two decimal places.

Mathematical Functions (Optional)

When you use a mathematical function, you must enclose the operand it is to work on in parentheses as follows:

```
mathematical-function (operand)
```

You can use the following mathematical functions:

Function	Description
ABS(operand)	Calculates the absolute value of the operand.
ATN(operand)	Calculates the arc tangent of the operand.
COS(operand)	Calculates the cosine of the operand.
EXP(operand)	Calculates the exponential of the operand.
FRAC(operand)	Calculates the fractional part of the operand.
INT(operand)	Calculates the integer part of the operand.
LOG(operand)	Calculates the Natural logarithm of the operand.
SGN(operand)	Calculates the sign of the operand (-1, 0, +1).
SIN(operand)	Calculates the sine of the operand.
SQRT(operand)	Calculates the square root of the operand. If the operand is negative, it is treated as positive. The maximum number of digits that can appear before the decimal point of the operand is 22.
TAN(operand)	Calculates the tangent of the operand.
VAL(operand)	Takes a numeric value from an alphanumeric operand and puts it into a numeric result field. The content of the operand must be the character representation of a numeric value. Leading or trailing blanks are ignored. Decimal points and leading sign characters are processed. If the result field is not long enough, decimal digits are truncated.
PERCENT(operand)	Calculates the percentage of the field for each record found in relation to the sum of all the values of the field found. For further information, see below.

Note:

The functions ATN, COS, LOG, SIN, and TAN first convert the operand to format F8, next evaluate the function, and then convert the result back to the original format of the operand.

PERCENT

The PERCENT function calculates the percentage of the field for each record found in relation to the sum of all the values of the field found.

Example:

```
RESULT = PERCENT (BUNKS)
```

If you use 'YACHT-NAME ST P' as your selection criteria, the field RESULT will show how many percent of the total number of bunks available each yacht whose name starts with P contains.

If you are dealing with percentages smaller than 1%, make sure that the field is defined with decimal places (for further information, see Working with the Worksheet).

Error 3088: Percent function not possible. Change processing sequence
 **To change the processing sequence**

1. Issue the OPTIONS command.

The Transaction Options window appears.

2. Page forward to Processing Options.
3. Change the option Processing Sequence so that S' comes before C'.
4. Press Enter.

You can now use the PERCENT function. For further information on transaction options, see Adjusting a User Profile.

Operands - Type and Format

Mathematical Function	Operand Type	Operand Format
ABS	Constant,	N P I F
ATN	Elementary field,	
COS	Single occurrence of	
EXP	multiple-value field	
FRAC	or periodic group	
INT		
LOG		
SGN		
SIN		
SQRT		
TAN		
VAL	Constant,	A
	Elementary field,	
	Single occurrence of	
	multiple-value field	
	or periodic group	
PERCENT	Elementary field,	N P I F
	Single occurrence of	
	multiple-value field	
	or periodic group	

Operators

Arithmetic Operators

The following operators are available:

Operand Function

+	Addition
-	Subtraction
*	Multiplication
/	Division
	Concatenation (joining fields)

Note:

If your keyboard does not have the vertical bar character (|), your system administrator will tell you which character to use.

Priority of Arithmetic Operators

You use parentheses to specify the order of operations. Operations enclosed within parentheses () will be performed first. The priority of operation processing is as follows:

Operations enclosed within parentheses ()

Multiplication and Division

Addition and Subtraction

If both multiplication and division, or both addition and subtraction, occur within an arithmetic calculation, the operation which occurs first has priority.

Examples of Arithmetic Operations:

```
SALARY = SALARY * 1.1
```

```
NEW-SALARY = SALARY * 1.1
```

```
PAST-EARNINGS = SALARY * YEARS-WITH-COMPANY
```

```
NEXT-5YR-EARNINGS = (SALARY * 1.25) * 5
```

```
NEXT-5YR-EARNINGS-PER-MONTH = (SALARY * 1.25) * 5 / 60
```

```
SEASON-LENGTH = SEASON-END - SEASON-START
```

Joining Fields (Concatenation)/Operator (|)

The joining fields/(concatenation) operator (|) is described in the section Different Types of Calculation Statements later in this section.

Note:

If your keyboard does not have the vertical bar character (|), your system administrator will tell you which character to use.

Different Types of Calculation Statements

There are three types of calculation statement as described in the following sections:

-

Arithmetic calculation

-

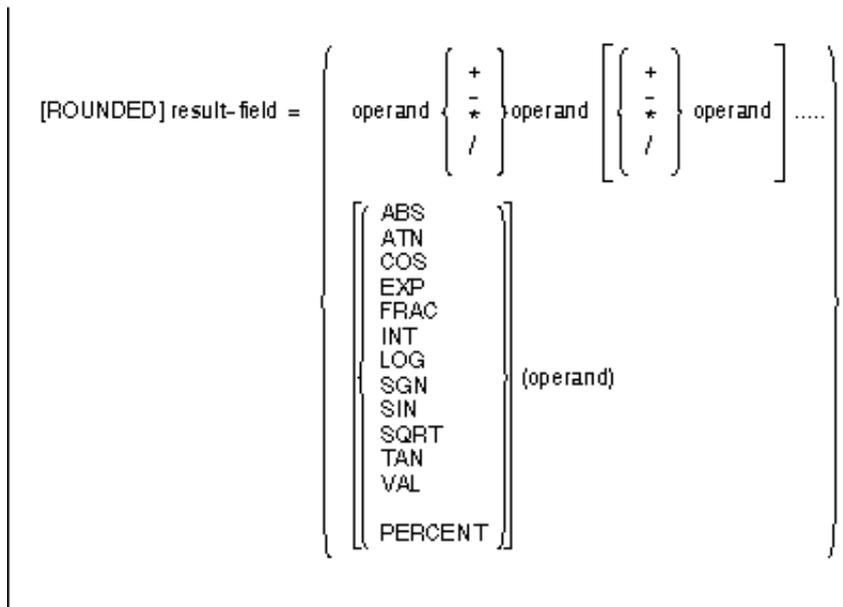
Assigning Values to a field

-

Joining fields (concatenation)

Arithmetic Calculation

You create arithmetic calculation statements as follows:

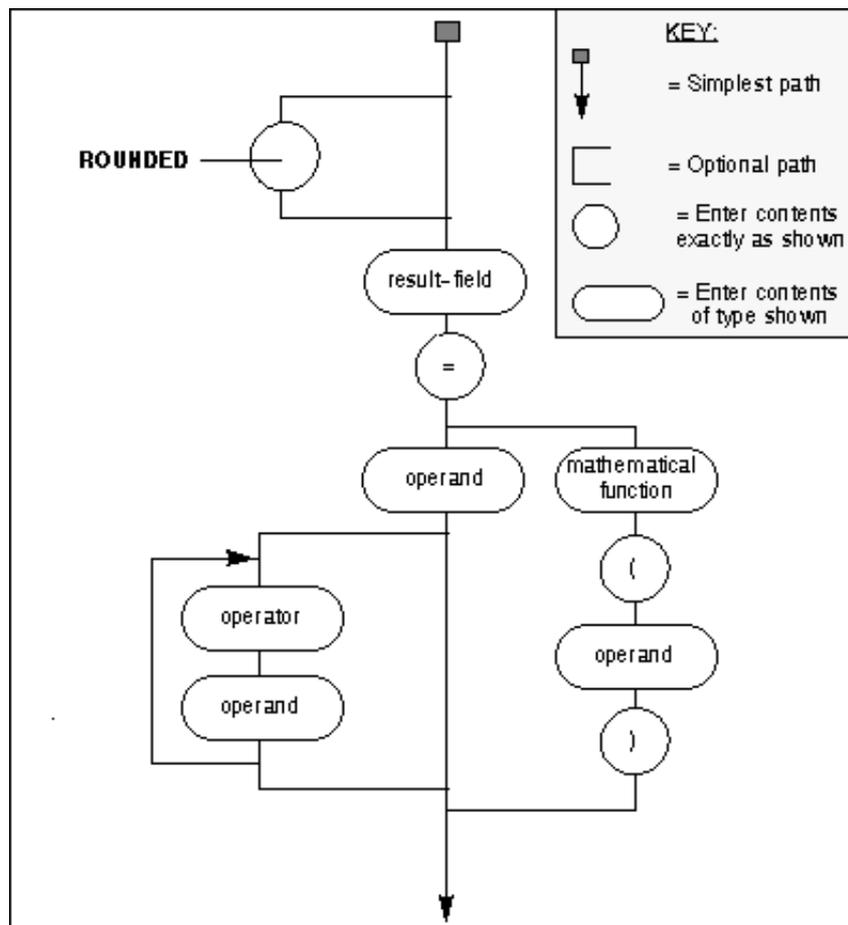


This documentation uses the Natural syntax conventions which are described in the Preface.

Note:

The Check and theContinue functions tell you exactly what you can enter in the situation you are in.

The following diagram uses an alternative method of representing arithmetic calculation statements:



You can use the following types of result field and operand:

Result Field	Operand
Numeric User Field	Numeric Elementary Field
Numeric Elementary Field	Numeric Multiple-value Field Numeric Periodic Group Numeric Constant Dynamic Input Field

For further information on working with multiple-value fields and periodic groups, see Array Processing.

Assigning Values to a Field

You can use the Calculation Editor to assign a new value to a field as shown below:

```
result-field = operand
```

The combination of result field and operand must be one of the following:

Result Field	=	Operand
Alphanumeric Elementary Field	=	Alphanumeric Constant
Alphanumeric Elementary Field	=	Numeric Constant
Alphanumeric Elementary Field	=	Alphanumeric Elementary Field
Alphanumeric Elementary Field	=	Numeric Elementary Field
Numeric Elementary Field	=	Numeric Constant
Numeric Elementary Field	=	Numeric Elementary Field
Exceptions:	=	
Numeric Elementary Field	=	Numeric Array(n-m)
Array1(n:m)	=	Array2(x:y)
Array1(n:m)	=	Constant

The value assigned has no effect on the actual value contained in the database, unless the assignment operation has been used in a data maintenance transaction.

Note:

Assignment values provided for alphanumeric fields must be enclosed in single quotation marks.

Examples of Assignment Operations

```
DID-BUSINESS-CONDITIONS = 'PAYED'
PRICE-1W = 300
```

Joining Fields (Concatenation)

You can use the joining fields (concatenation) operator to combine the values of several fields and/or user-specified text into a single output field. Use the vertical bar (|) (EBCDIC H4F') as shown below:

```
result-field = operand | operand | [ operand ] ...
```

The result field must be an alphanumeric field. The operand can be an elementary field, a multiple-value field, periodic group, hex constant, dynamic input field, text string (alphanumeric constant) or blank.

You can use the following types of result field and operand:

Result Field	Operand
Alphanumeric Field	Alphanumeric Elementary Field Alphanumeric Multiple-value Field Periodic Group Hex Constant Dynamic Input Field Text String (Alphanumeric Constant) Blank

Any text or blanks that are to be inserted between the field values must be enclosed between single quotation marks. Any hexadecimal values to be inserted must be enclosed between single quotation marks and preceded by H.

Note:

If your keyboard does not have the vertical bar character, your system administrator will tell you which character to use.

Example:

```
RESULT-FIELD = START-HARBOR | START-DATE
```

results in the field RESULT-FIELD having the following contents:

```
MARMARIS19910729
```

where MARMARIS is the current value of the field START-HARBOR and 19910729 is the current value of the field START-DATE.

Example:

```
RESULT-FIELD = SURNAME | ' , ' | FIRST-NAME-1
```

results in the field RESULT-FIELD having the following contents:

```
BAKER , PAULINE
```

where BAKER is the current value of the field SURNAME and PAULINE is the current value of the field FIRST-NAME-1.

Arithmetic Operations with Date and Time

With formats D (date) and T (time), only addition and subtraction are allowed; multiplication and division are not allowed.

Date/time values can be added to/subtracted from one another; or integer values (no decimal digits) can be added to/subtracted from date/time values. Such integer values can be contained in fields of formats N, P, I, D, or T.

An integer value added to/subtracted from a date value is assumed to be in days. An integer value added to/subtracted from a time value is assumed to be in tenths of seconds.

For arithmetic operations with date and time, certain restrictions apply, which are due to the Natural's internal handling of arithmetic operations with date and time, as explained below.

Internally, Natural handles an arithmetic operation with date/time variables as follows:

```
COMPUTE result-field= operand1+/-operand2
```

The above statement is resolved as:

- intermediate-result=operand1+/-operand2
- result-field=intermediate-result

That is, in a first step Natural computes the result of the addition/subtraction, and in a second step assigns this result to the result field.

More complex arithmetic operations are resolved following the same pattern:

```
COMPUTE result-field = operand1+/- operand2+/- operand3+/- operand4
```

The above statement is resolved as:

- intermediate-result1=operand1+/-operand2
- intermediate-result2=intermediate-result1+/-operand3
- intermediate-result3=intermediate-result2+/-operand4
- result-field=intermediate-result3

The internal format of such an intermediate-result depends on the formats of the operands, as shown in the tables below.

In the tables below, *Di* is a value in internal date format; *Ti* is a value in internal time format; such values can be used in further arithmetic date/time operations, but they cannot be assigned to a result field of format D (see the assignment table below).

In complex arithmetic operations in which an intermediate result of internal format *Di* or *Ti* is used as operand in a further addition/subtraction, its format is assumed to be D or T respectively.

The following table shows the format of the intermediate-result of an addition (intermediate-result=operand1+operand2):

Format of operand1	Format of operand2	Format of intermediate-result
D	D	Di
D	T	T
D	N P I	D
T	D T N P I	T
N P I	D	D
N P I	T	T

The following table shows the format of the intermediate-result of a subtraction (intermediate-result=operand1-operand2):

Format of operand1	Format of operand2	Format of intermediate-result
D	D	Di
D	T	Ti
D	N P I	D
T	D T	Ti
N P I	D	Di
N P I	T	Ti

The following table shows which intermediate results can internally be assigned to which result fields (result-field=intermediate-result).

Format of result-field	Format of intermediate-result	Format of possible
D	D T	yes
D	Di Ti N P I	no
T	D T Di Ti N P I	yes
N P I	D T Di Ti N P I	yes

A result field of format D or T must not contain a negative value.

Examples 1 and 2 (invalid):

```
COMPUTE DATE1 (D) = DATE2 (D) + DATE3 (D)
```

```
COMPUTE DATE1 (D) = DATE2 (D) - DATE3 (D)
```

These operations are not possible, because the intermediate result of the addition/ subtraction would be format *Di*, and a value of format *Di* cannot be assigned to a result field of format D.

Examples 3 and 4 (invalid):

```
COMPUTE DATE1 (D) = TIME2 (T) - TIME3 (T)
```

```
COMPUTE DATE1 (D) = DATE2 (D) - TIME3 (T)
```

These operations are not possible, because the intermediate result of the addition/ subtraction would be format *T i*, and a value of format *T i* cannot be assigned to a result field of format D.

Example 5 (valid):

```
COMPUTE DATE1 (D) = DATE2 (D) - DATE3 (D) + TIME3 (T)
```

This operation is possible. First, DATE3 is subtracted from DATE2, giving an intermediate result of format *Di*; then, this intermediate result is added to TIME3, giving an intermediate result of format *T*; finally, this second intermediate result is assigned to the result field DATE1.

If a format *T* value is assigned to a format *D* field, you must ensure that the time value contains a valid date component.

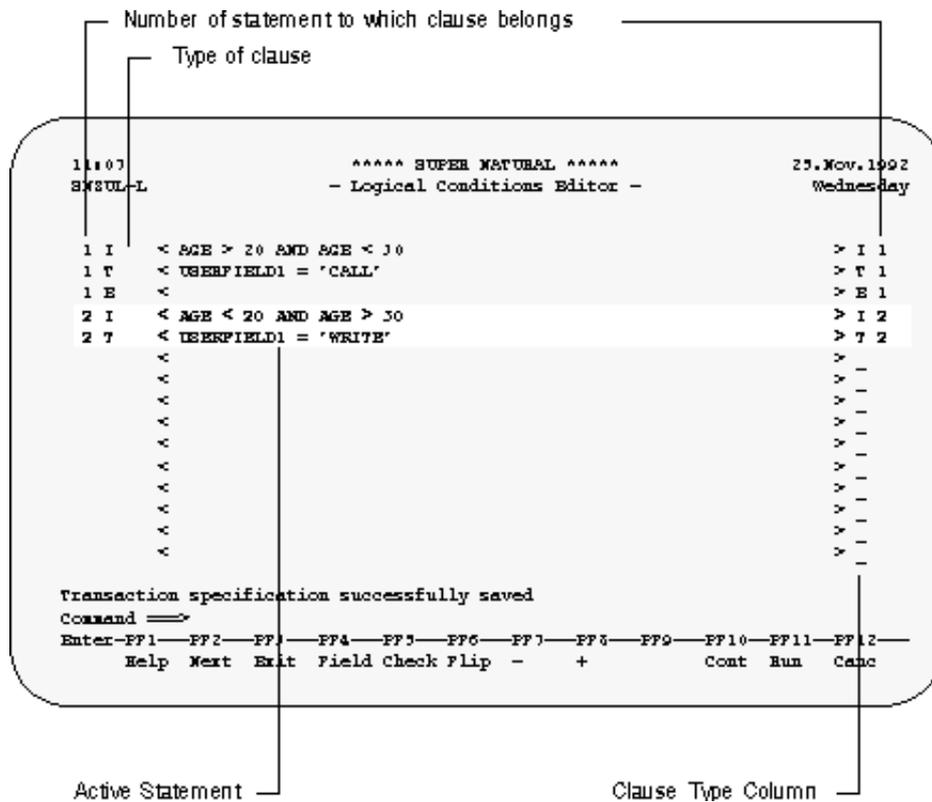
The Logical Conditions Editor

You use the Logical Conditions Editor to define logical condition statements using IF, THEN and ELSE clauses. You can define multiple logical condition statements.

To use the Logical Conditions Editor:

- Issue the LOGICcommand.

The Logical Conditions Editor appears:



There are three types of lines in the Logical Conditions Editor. Entries made in the I lines are interpreted as IF clauses, entries made in the T lines are interpreted as THEN clauses and entries made in the E lines are interpreted as ELSE clauses. Every logical condition statement must start with an IF clause.

You can only edit the active statement. The active statement is highlighted and the rest of the editor area is protected. If you want to edit another statement, you must activate it first.

▶ To activate a statement:

1. Position the cursor in the statement you want to activate.
2. Press Enter.

The statement is now highlighted and you can edit it.

When you invoke the Logical Conditions Editor, one line of each type is available. If a clause is longer than one line, you must create a new line of the same type for the remainder.

▶ To create a new line of same type:

- Use the .I or .Ceditor commands as described earlier in this section.

▶ To change the line type:

1. Position the cursor in the Clause Type column next to the line whose type you want to change.
2. Overtyping the present entry with the letter of the required clause type.
3. Press Enter.

► **To start a new statement:**

1. Activate the last statement.
2. Enter an "I" in the Clause Type column below the last entry.

Every logical condition statement must start with an IF clause.

A line for the IF clause of a new statement is created and numbered accordingly.

A new statement is only started if the previous statement contains a THEN or an ELSE line.

The Continue Function

As in the Selection and Calculation Editors, you use theContinue function to obtain information on what you can enter next.

The Continue function in the Logical Conditions editor differs from the Continue function in the other full-screen editors in the following cases:



When writing the first statement

If the IF or THEN clause is complete as it stands, you get information on the next clause type. If you want information on the same clause type, enter any non-blank character before invoking the Continuefunction.



When the statement(s) contain errors

TheCONTINUEfunction gives you information on how to correct the errors.

If you want information on what you can enter in the next line type in the second and subsequent statements, you must create a blank line of that type before invoking the Continue function.

Logical Condition Statements

You can define logical condition clauses in the following combinations:

Condition Combination	Interpretation
IF x THEN y	If condition x is fulfilled, then do y.
IF x THEN y ELSE z	If condition x is fulfilled, then do y, otherwise do z.
IF x ELSE z	Ignore records where condition x is fulfilled, for all other records do z.

For information on using dynamic input fields, see Operators in **Selection Editor**.

IF Clause

IF clauses are constructed the same way as selection statements with the following exceptions:



You can use user fields and dynamic input fields as well as database fields to start the statement.

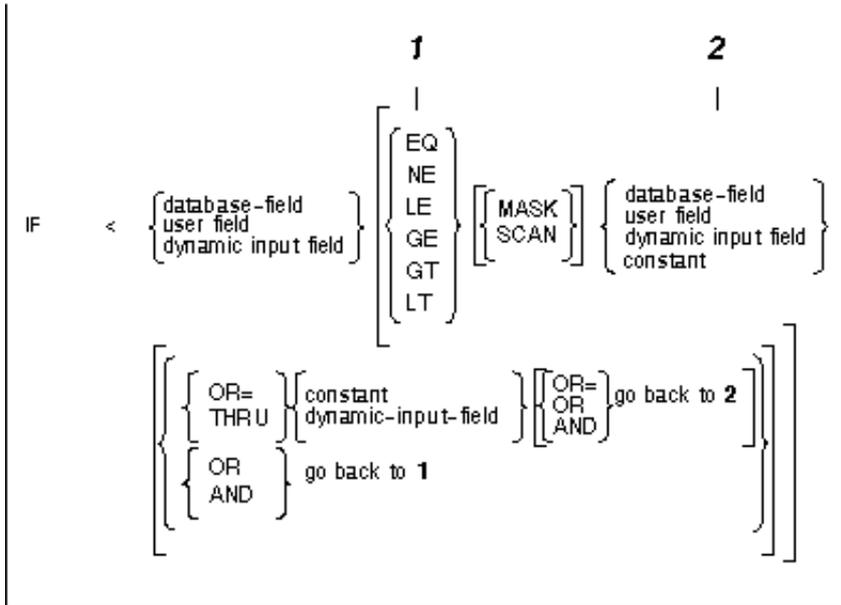


You may not use the START operator.

-

You can use user fields and database fields as well as dynamic input fields and constants after an operator.

You create IF clauses as follows:



This documentation uses the Natural syntax conventions which are described in the Preface.

Note:

The Continue function tells you exactly what you can enter in the situation you are in.

For further information on selection statements, see The Selection Editor .

THEN Clause

THEN clauses are constructed the same way as calculation statements with the following exceptions:

-

You can use the ACCEPT and REJECT keywords

-

You can use the STOP keyword.

For further information on calculation statements, see The Calculation Editor.

ACCEPT and REJECT Keywords

You use the ACCEPT and REJECT keywords as follows:



Note:

If you use ACCEPT or REJECT, you may use nothing else in the THEN clause and you may not define an ELSE clause after it.

Condition	Interpretation
IF x THEN ACCEPT	Accept all records where condition x is fulfilled
IF x THEN REJECT	Reject all records where condition x is fulfilled

Note:

You can use fields from the lookup file with the ACCEPT and REJECT keywords.

STOP Keyword

You use the STOP keyword as follows:

```
I < x T < STOP
```

Note:

If you use the STOP keyword, you may use nothing else in the THEN clause and you may not define an ELSE clause after it.

Condition	Interpretation
IF x THEN STOP	If condition x is fulfilled, stop processing.

ELSE Clause

The ELSE statement is the same as the calculation statement.

For further information on calculation statements, see The Calculation Editor.

Note:

The Continue function tells you exactly what you can enter in the situation you are in.

The SQL SELECT Editor

You use the SQL SELECT Editor for entering SQL (Structured Query Language) SELECT statements for reporting transactions with the reporting mode Data Selection SQL-SELECT. (For further information on reporting modes, see Defining Transaction Modes for Reporting). You must enter selection criteria using SQL (Structured Query Language) SELECT statements. Selection criteria tell Super Natural which data you want from the database.

Users of the SQL SELECT Editor must already be familiar with SQL.

Note:

The CONTINUE command is not available in the SQL SELECT Editor.

▶ To use the SQL SELECT Editor:

1. Issue the SELECTION command.

The SQL Editor appears:

```

15:41                      ***** Super Natural *****                05.Jan.1993
SNZUL-Q                    - SQL SELECT Editor -                          Tuesday

      1 < SELECT ENAME, CITY, AGE, SEX                                     > 1
      2 < INTO ENAME, CITY                                               > 2
      3 < FROM EZA-EMPLOYEES                                             > 3
      4 <                                                                 > 4
      5 <                                                                 > 5
      6 <                                                                 > 6
      7 <                                                                 > 7
      8 <                                                                 > 8
      9 <                                                                 > 9
     10 <                                                                 > 10
     11 <                                                                 > 11
     12 <                                                                 > 12
     13 <                                                                 > 13
     14 <                                                                 > 14
     15 <                                                                 > 15

NAT0673 Number of SELECT-list and INTO-list elements do not match.
Command ===>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Next Exit Field Check Flip - + Table Run Canc

```

2. Enter your selection criteria using SQL SELECT statements.

You can use the following types of clause in the SQL SELECT Editor:

- SELECT (mandatory)
- INTO (mandatory)
- FROM (mandatory)
- WHERE (optional)
- GROUP BY (optional)
- HAVING (optional)
- ORDER BY (optional)
- UNION (optional)

For information on the optional clauses, see *Natural Reference documentation* or other SQL documentation.

The following sections describe how to create the SELECT, INTO and FROM clauses covering aspects specific to Super Natural.

SELECT Clause

The SELECT clause is used to define output columns for the result table and can contain fields from DB2 tables, text strings and arithmetic expressions.

You can either enter the names of the fields you want to select directly, or you can use the TABLES function to help you.

Note:

You must separate column names with a comma (,).

To enter fields directly:

- Enter the names of the fields you want use.

To enter fields using the TABLES command:

1. Issue the TABLES command to obtain a list of the DB2 tables you are authorized to use.

The File Selection List window appears listing the DB2 tables available.

2. Select a DB2 table.

The Field Selection List window appears showing the fields available in the DB2 table.

3. Select the fields you want to use in the SELECT clause.

At this point, you can also choose to automatically generate user fields to match the DB2 fields you have selected as described below.

The user fields have the same name, format and length as their counterpart and can be used in the INTO clause.

To automatically generate user fields to match the DB2 fields you have selected in the Field Selection List window:

1. Mark the Generate field provided in the Field Selection List window.
2. Press Enter.

The field names appear in the SELECT clause line and if you have marked the Generate field, user fields are generated to match and appear in the Fieldname column in the Worksheet.

INTO Clause

Every output column defined in the SELECT clause must have a user field of like format and length in the INTO clause. Only the first SELECT clause needs a corresponding INTO clause.

Note:

You must separate user field names with a comma (,).

You can either generate the user fields automatically as described above or enter the names directly. If you enter the name of a non-defined field, you must define it later in the Define User Field window which appears when you issue the CHECK command or leave the SQL Editor with EXIT. You can also define user fields from the Worksheet.

▶ **To obtain a list of the user fields you can use:**

- Issue the FIELDS command.

A selection list of the user fields which are already in the Fieldname column of the Worksheet appears. The field list is empty if no user fields are defined.

For information on the interpretation of null values, see the section **INDICATOR/LINDICATOR** in the description of SQL SELECT statements in the *Natural Reference documentation*.

FROM Clause

The name of every DB2 table which you want to use in the SELECT statement must appear in the FROM clause. You can use the TABLES function to remind you of the DB2 tables available but you can not use it to enter the table names in the editor for you.

Issuing the TABLES and FIELDS Commands

Before you issue the TABLES or FIELDS command, you must press Enter and position the cursor to where fields can be inserted using the arrow keys (not the space bar). You can issue the TABLES and FIELDS commands either by entering them in the command line or by pressing the relevant PF key.

Note:

The FIELDS command is allocated to PF4. The TABLES command is allocated to PF10.

SQL Help Routine

As well as the general editor help available in all editors, the SQL SELECT Editor has its own SQL-specific help routine which gives information on each of the types of clause available.

▶ **To access the SQL Help Routine:**

1. Position the cursor inside the editor area.
2. Press PF1.

The menu for the SQL Help Routine window appears:

```

15:42                ***** Super Natural *****                05.Jan.1993
SHSQL                - SQL Select - Help Routine -                Tuesday

      +-----+          +-----+          +-----+
      I SELECT-Clause I      I FROM-Clause I      I WHERE-Clause   I
      +-----+          +-----+          +-----+
      +-----+          +-----+          +-----+
      I INTO-Clause   I      +-----+          I GROUP BY -Clause I
      +-----+          +-----+          +-----+
                                          +-----+
                                          I HAVING-Clause   I
                                          +-----+
                                          +-----+
                                          I ORDER BY -Clause I
                                          +-----+
                                          +-----+
                                          I UNION-Clause   I
                                          +-----+

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
                Exit                Flip                Canc

```

Each section in the SQL Help Routinewindow represents a help sub-system.

 **To access a help sub-system:**

1. Position the cursor at the sub-system you require.

Use the carriage return key or tabs to do this, not the arrow keys.

2. Press Enter.

Marking options takes you to further help windows, theCANCEL command takes you to the last window and theEXITcommand you takes you back to the SQL- SELECT Editor from any point within the SQL help routine.

The PC File Description Editor

When you are adding a reporting transaction with the transaction modeData Selection set to DATA FROM PC FILE, you must use the PC File Description Editor to describe the file layout of the file you are using.

 **To invoke the PC File Description Editor:**

1. Issue the SELECTION command.

```

15:44          ***** Super Natural *****          05.Jan.1993
SNZUL-WF          - PC File Description Editor -          Tuesday

      Fieldname          Format Length Dec.Places
1 <          >
2 <          >
3 <          >
4 <          >
5 <          >
6 <          >
7 <          >
8 <          >
9 <          >
10 <         >
11 <         >
12 <         >
13 <         >
14 <         >
15 <         >

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Next  Exit          Check Flip -      +          Run  Canc

```

2. Enter the name of the fields to be read from the PC file, their format and length in the relevant columns.

To prevent errors when the transaction is run, the following restrictions are applied to the editor area as soon as one or more fields are used elsewhere in the transaction:

- You cannot enter new fields
- You cannot delete fields
- You cannot change the field order
- The fields used elsewhere in the transaction are locked
- Fields only used in the editor can only be edited

The availability of line commands and editor direct commands are restricted accordingly.

Note:

To lift the restrictions, remove all fields from use in the transaction, for example from the Worksheet, any other editor or in a prototype program. The Info column in the Worksheet provides you with information on field usage.

The Work File Description Editor

When you are adding a reporting transaction with the transaction modeData Selection set to DATA FROM WORK FILE, you must use the Work File Description Editor to describe the file layout of the file you are using.

To invoke the Work File Description Editor:

- Issue the SELECTION command.

```

15:45                ***** Super Natural *****                05.Jan.1993
SNZUL-WF              - Work File Description Editor -                Tuesday

      Fieldname                Format  Length  Dec.Places
1  <                            >
2  <                            >
3  <                            >
4  <                            >
5  <                            >
6  <                            >
7  <                            >
8  <                            >
9  <                            >
10 <                            >
11 <                            >
12 <                            >
13 <                            >
14 <                            >
15 <                            >

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Next  Exit      Check Flip  -    +                Run  Canc

```

Enter the name of the fields to be read from the work file, their format and length in the relevant columns. To prevent errors when the transaction is run, the following restrictions are applied to the editor area as soon as one or more fields are used elsewhere in the transaction:

- You cannot enter new fields
- You cannot delete fields
- You cannot change the field order
- The fields used elsewhere in the transaction are locked
- Fields only used in the editor can only be edited

The availability of line commands and editor direct commands are restricted accordingly.

Note:

To lift the restrictions, remove all fields from use in the transaction, for example from the Worksheet, any other editor or in a prototype program. The Infocolumn in the Worksheet provides you with information on field usage.

The User File Description Editor

When you are performing the Add Userfilefunction, you use the User File Description Editor to describe the layout of the user file you are creating by adding, modifying and/or deleting field descriptions. For further information on the User File Description Editor, see Working with User Files.

Arrays

Referencing Occurrences/Arrays

Multiple-Value Fields and Periodic Groups

When you need to deal with a particular occurrence of a multiple-value field or a periodic group, you must give it a reference number or index as shown below:

<pre>fieldname { (occurrence-no.-periodic-group) (occurrence-no.-multiple-value-field) }</pre>
--

Example:

The following refers to the first occurrence of the multiple-value field EXTRAS in the demo file E-SAG-TOURS-CH-PRICES:

```
EXTRAS ( 1 )
```

Multiple-Value Field or Periodic Group Within a Periodic-Group

Two reference numbers are required to reference an occurrence of a multiple-value field within a periodic group as shown below:

<pre>fieldname(occurrence-no.-PG,occurrence-no.-MU)</pre>

where PG is periodic group and MU is multiple-value field.

The first number indicates which occurrence of the periodic group it is in and the second number indicates which occurrence of the multiple-value field it is. Separate the reference numbers with a comma (,) and enclose them within parentheses(()).

Example:

The following refers to the third occurrence of the multiple-value field LANGUAGES, contained in the second occurrence of the periodic group SAILOR in the demo file E-SAG-TOURS-CRUISE:

LANGUAGES (2 , 3)

For further information on multiple-value fields and periodic groups, see **Field Types** in **Defining Fields**.

Array Processing

All scalar operations can be applied to array elements which consist of a single occurrence.

Assignment Operations with Arrays

If an array range is assigned to another array range, the assignment is performed element by element.

If a single occurrence is assigned to an array range, each element of the range is filled with the value of the single occurrence. (For a mathematical function, each element of the range is filled with the result of the function.)

Before an assignment operation is executed, the individual dimensions of the arrays involved are compared with one another to check if they meet one of the conditions listed below. The dimensions are compared independently of one another; that is, the 1st dimension of the one array is compared with the 1st dimension of the other array and the 2nd dimension of the one array is compared with the 2nd dimension of the other array.

The assignment of values from one array to another is only allowed under one of the following conditions:

-

The number of occurrences is the same for both dimensions compared.

-

The dimension that is assigned to another dimension consists of a single occurrence.

Sample Array Assignments:

```
FIELD1(1:3) = FIELD2(6:8)
FIELD3(1,1:3) = FIELD4(6:8)
FIELD5(1:3,3:5) = FIELD6(1:3,1:3)
```

Comparison Operations with Arrays

Generally, the following applies: if arrays with multiple dimensions are compared, the individual dimensions are handled independently of one another. The 1st dimension of the one array is compared with the 1st dimension of the other array and the 2nd dimension of the one array is compared with the 2nd dimension of the other array.

The comparison of two array dimensions is only allowed under one of the following conditions:

-

The array dimensions compared with one another have the same number of occurrences.

-

The array dimensions compared with one another have an indefinite number of occurrences.

Note:

Only in the IF clause of the Logical Conditions editor.

-

All array dimensions of one of the arrays involved are single occurrences.

The following sample shows which array comparison operations are possible:

Sample Array Comparisons:

```
IF A2(1,1) = A1(1) THEN ACCEPT
IF A2(1,*) = A1(1) THEN STOP
IF A2(1,*) = A1(*) THEN ACCEPT
IF A2(1,1) = A2(1,1) THEN REJECT
```

Comparing Two Array Ranges:

Example:

```
IF #ARRAY1(1:2) NE #ARRAY2(1:2)
```

The above condition is fulfilled if the first occurrence of #ARRAY1 does not equal the first occurrence of #ARRAY2 *and* the second occurrence of #ARRAY1 does not equal the second occurrence of #ARRAY2.

This is equivalent to the following expression:

```
IF (#ARRAY1(1) NE #ARRAY2(1)) AND (#ARRAY1(2) NE #ARRAY2(2))
```

Arithmetic Operations with Arrays

Array ranges can be specified in the following ways:

-

ADDrangeTOrangeGIVINGrange

Range dimensions must be equal. ADD is performed element by element.

-

MULTIPLYrangeBYrangeGIVINGrange

Range dimensions must be equal. MULTIPLY is performed element by element.

-

ADDscalarTOrangeGIVINGrange

Range dimensions must be equal. Scalar is added to each element of range.

-

MULTIPLYrangeBYscalarGIVINGrange

Range dimensions must be equal. Each element of range is multiplied by scalar.

-

ADDrangeTOscalarGIVINGscalar

Each element of the array is added to scalar and the result is stored in scalar.

-

MULTIPLYscalarBYrangeGIVINGscalar2

Scalar is multiplied by each element of the array and the result is assigned to scalar2.