Select.Seek Subroutine

Description

Select.Seek is used to change the current position in a select list.

Syntax

Select.Seek*(mode, cursor, position, flag)*

Using Select.Seek

Use Select.Seek to move either absolutely or relatively within a select list. You can go directly to the top, bottom, or specific place in a select list, or you can go to a position that is relative to one of those absolute locations. For example, you can go to the top of to 5 positions on either side of the top.

The select list may be obtained either through Basic+ (Select or Select...By) or by the RList subroutine. The select list may be either latent or resolved.

When using Select.Seek, make note of the following requirements:

- You may execute either a standard or an extended Select process, but you must use the Readnext By command, with a "non-terminating" option. The selections may be either ascending or descending.
- If you are using a standard Select process, ou must still use the extended Readnext By command to read record keys. In that case, use 0 as the cursor number.

You may specify relative positions by using *position*. For example, the code:

```
Select.Seek(current_seek$, 0, 5, flag)
```

will take you five positions past your current position. Similarly,

```
Select.Seek(current_seek$, 0, 5, flag)
```

will take you five positions back from the current position. Also,

```
Select.Seek(bottom_seek$, 0, -5, flag)
```

will take you to the fifth position before the end of the Select list.

You may also mark a particular position, do some processing that reposition the select list, then return to the marked position.

An error code is returned in *flag*.

Parameters

The Select.Seek subroutine has the following parameters:

Parameter	Description

<table>
<tr><td>mode</td><td>Mode determines either the position in the select list (top, bottom, current) from which to calculate the new position (top+0, top+5, current-1), or it sets a marker so that you can later restore the current position.

There are five possible arguments for mode:

<table>
<tr><th>Argument</th><th>Meaning</th></tr>
<tr><td>0</td><td>Begin from the top</td></tr>
<tr><td>1</td><td>Begin from the bottom</td></tr>
<tr><td>2</td><td>Begin from the current position</td></tr>
<tr><td>3</td><td>Mark the current position</td></tr>
<tr><td>4</td><td>Return to the previously marked position</td></tr>
</table>

This argument must be supplied, and it must be numeric.</td></tr>
<tr><td>cursor</td><td>An argument for the cursor parameter must be a value from 0 through 8. This argument must be supplied, and it must be numeric.

• If you selected record keys using an Basic+ or RList Select statement, cursor must be 0.</td></tr>
<tr><td>position</td><td>Position has two functions, depending on mode:

<table>
<tr><th>When mode is</th><th>Meaning of position</th></tr>
<tr><td>0</td><td>Offset value</td></tr>
<tr><td>1</td><td>Offset value</td></tr>
<tr><td>2</td><td>Offset value</td></tr>
<tr><td>3</td><td>Return variable</td></tr>
<tr><td>4</td><td>Position data</td></tr>
</table>

When the value of position is an offset value, it specifies how far ahead (a positive value) or behind (a negative value) the default value a location is to be. For example, if the top_seek argument is specified (mode 0), an argument for position will position the select list either ahead or behind the top position an amount that is the value of position. To go to an absolute top, bottom, or current position, make position null or zero.

When marking or restoring the select list position, position contains the positioning data needed by Select.Seek to restore the select position. Therefore, position returns a value when executing the mark_seek command (mode 3), then supplies that value for the restore_seek process (mode 4).</td></tr>
<tr><td>flag</td><td>Pass an argument for flag to return true or false, depending on the success of the operation.</td></tr>
</table>

Values Returned

Flag returns true if the process succeeded, false if it did not.

Position returns data specifying the current position after a **mark_seek** operation.

Example

```
/* The following code opens the SAMPLE_CUSTOMERS file, runs Select on it, then exercises various Select.Seek
functions. */


Declare subroutine Select.Seek, FsMsg, Msg


Equ top_seek$     to 0
Equ bottom_seek$  to 1
Equ current_seek$ to 2
Equ mark_seek$    to 3
Equ restore_seek$ to 4


filename  = "SAMPLE_CUSTOMERS"
```

```
fieldname = "COMPANY_NAME"
cursor    = 0
position  = ""


ClearSelect    ; // Just in case


Open filename to filevar else FsMsg(); Stop
Select filevar


// Move to the first record key in the select list
mode  = top_seek$
place = "top"
GoSub go_seek   ; // set up arguments for a top_seek


// move to the last record key in the select list
mode  = bottom_seek$
place = "bottom"
GoSub go_seek


// mark the current position in the select list
mode  = mark_seek$
place = "bottom"
GoSub go_seek


// move to fifth record key from the bottom of the select list
mode     = bottom_seek$
position = -5
place = position : " from bottom"
GoSub go_seek


// return to the position marked above
mode     = restore_seek$
position = store_position  ; // restore the marked position
Select.Seek(mode, cursor, position, flag)
If flag else Stop
Msg(@Window, "Here is restored ID: " : @ID)


Stop ; // end of program


**********************
* Internal Subroutines
**********************


go_seek:
    Select.Seek(mode, cursor, position, flag)
    If flag then
        Readnext @ID using cursor By AN else
            FsMsg()
            Stop
        end
        Msg(@Window, "Seek to the " : place : " ID: " : @ID)
    end else
        Msg(@Window, "Seek to " : place : " failed")
    end
return
```