Select...By Statement

Description

An extended version of the Select statement which supports multiple cursors and sort options. The Select...By statement results in a resolved (not latent) cursor.

Syntax

Select *table* **By** *column keyword cursorvar* Then | Else statements

Parameters

The Select...By statement has the following parameters.

Parameter	Description
Table	Designates the table that is to be accessed by the Select statement. Unlike the basic Select syntax, this table name is provided as a literal, not as a table variable generated by an Open statement.
Sort	Used to specify the sort order for the keys to be returned in the cursor. The parameter sort consists of a field mark-delimited array of column names. The column names must resolve to the names of columns in the table represented by table. The keys in the cursor will be sorted according to the field(s) in sort. The default sort order for any column specified in sort is an ascending sort. To indicate a descending sort for any column, precede the column name with a pound sign. For example, to indicate reverse sort order for the COMPANY column, use the expression #COMPANY. Sort priority is determined by the order of the columns in the list. The first element in the list will be the primary sort criterion, the second element the secondary criterion, etc. If you do not wish to specify a sort order for keys in the cursor, but still wish to use the extended syntax for Select, specify a null ("") for sort.
Keyword	Determines the disposition of the cursor after Select...By has sorted the key list. There are four possible keywords: {KEYWORD_TABLE}
Cursorvar	Identifies the cursor to use. The Select...By command initializes this variable with information used for subsequent access to the cursor if the keyword setting is used.
Then	The statement(s) following Then are executed if the cursor is properly initialized.
Else	The statement(s) following Else are executed if the Select...By operation is not successful. The Status() function indicates the severity of the error, and the system variable @FILE_ERROR contains detail about the nature of the error.

Keyword table (embedded within the *Keyword* row):

Keyword	Meaning
Setting	Find the next available cursor, establish the select list in that cursor, and return the cursor number in cursorvar. Setting will return a false branch (Else) if no cursor is available.
Using	Allows resorting an active list by specifying new sort criteria in sort. The cursor specified in Using must have been assigned the same table as that of table. If the cursor has been set up for a table, and then a Select...By is performed on that cursor using a different table, the cursor will be cleared and reset. All previous sort and selection criteria are lost.
Assigning	Allows the key list in a cursor to be assigned to a table name different from the one used to generate the key list. Assigning thus functions like Using, but permits the list in a cursor to be resorted and applied to a different table.
Mode variable, cursorvar	Allows you to set the value of Keyword at run-time instead of hard-coding it into the Select...By syntax. When using Mode, supply the name of a variable, which, during program execution, you must assign an integer value as follows. · Differs from the other keywords in that it permits the designation of a cursor keyword (Setting, Using, or Assigning) by means of a variable. This allows a program to determine the proper keyword for Select at program execution time. · The value of variable must be an integer designating the proper keyword. The values to use for each keyword are: { Keyword / Value — Setting 1, Using 2, Assigning 3 }

Keyword	Value
Setting	1
Using	2
Assigning	3

See also

Select, Reduce, RList

Example

```
/* This program fragment creates a select list out of rows in the CUSTOMER file. The list is sorted according
to the columns ZIP and COMPANY. The resultant select list will be assigned to the next available Cursor. The
Cursor number for the Cursor assigned by Setting is passed back to the program in Cursor. If no Cursors are
available, or if the sort is not successful, the program returns an error. */
sort_list = "ZIP" : @FM : "COMPANY"
Select "CUSTOMER" By sort_list Setting Cursor Else
       status = Set_FSError()
       Return
End
/* This program fragment illustrates the use of the Assigning keyword. In this example, the first Select...By
establishes a select list (in descending order by salary) of rows in the EMPLOYEE.MASTER file. The second
Select...By statement assigns the BENEFITS file to this Cursor. This permits the processing loop to use the
keys from the EMPLOYEE.MASTER file to read records from the BENEFITS file. In this example, it is assumed that
the files share a common key structure. */
* select employees by salary (highest to lowest)
Select "EMPLOYEE_MASTER" By "#SALARY" Using 1 Else
       status = Set_FSError()
       Return
End
* assign that list to a new table
Select "BENEFITS" By "" Assigning 1 Else
       status = Set_FSError()
       Return
End

Done = 0
Loop
       ReadNext @ID Using 1 By At Else Done = 1
Until Done DO
       GoSub PROCESS_RECORD
Repeat
```