

Copy_Row Subroutine

Description

Copies a row or a group of rows from one table to another.

Syntax

Copy_Row (*scrtablename*, *scrkeylist*, *desttablename*, *destkeylist*, *overwriteflag*, *lockflag*, *deleteSource*)

Parameters

The Copy_Row subroutine has the following parameters.

Parameter	Description								
<i>scrtablename</i>	The name of the table that contains the row or rows to be copied.								
<i>scrkeylist</i>	A key or list of keys to copy from scrtablename to desttablename. A one-to-one correspondence must exist between scrkeylist and destkeylist. Note: If more than one rows are to be copied, separate the keys with field marks. See the Example 2 below.								
<i>desttablename</i>	Specifies the name of the table into which the rows are copied. If null, then scrtablename is used.								
<i>destkeylist</i>	A key or list of keys to copy from scrtablename to the keys specified in destkeylist in the desttablename table. A one-to-one correspondence must exist between destkeylist and scrkeylist. If null, then scrkeylist is used.								
<i>overwriteflag</i>	Specifies whether or not rows copied to a new table overwrite an existing row. The default value is 0 (no overwrites). <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>Prevents overwriting an existing row (default).</td></tr><tr><td>1</td><td>Only replace an existing row. If destination row does not exist, then skip this row.</td></tr><tr><td>2</td><td>Write this row, unconditionally. If the destination row exists, overwrite it.</td></tr></table>	Value	Description	0	Prevents overwriting an existing row (default).	1	Only replace an existing row. If destination row does not exist, then skip this row.	2	Write this row, unconditionally. If the destination row exists, overwrite it.
Value	Description								
0	Prevents overwriting an existing row (default).								
1	Only replace an existing row. If destination row does not exist, then skip this row.								
2	Write this row, unconditionally. If the destination row exists, overwrite it.								
<i>lockflag</i>	Specifies whether rows are locked during the copy process. The default is 0 (no locking). <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>No locking during the copy process.</td></tr><tr><td>1</td><td>Lock the source & destination rows during the copy process.</td></tr></table>	Value	Description	0	No locking during the copy process.	1	Lock the source & destination rows during the copy process.		
Value	Description								
0	No locking during the copy process.								
1	Lock the source & destination rows during the copy process.								
<i>deleteSource</i>	Specifies whether the source rows are deleted during the copy process. The default is 0 (no deletion). <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>The source rows are not deleted after copying.</td></tr><tr><td>1</td><td>The source rows are deleted after copying.</td></tr></table>	Value	Description	0	The source rows are not deleted after copying.	1	The source rows are deleted after copying.		
Value	Description								
0	The source rows are not deleted after copying.								
1	The source rows are deleted after copying.								

Notes: Although the Copy_Row subroutine can be called from a BASIC+ program, it is preferable to use the [Read](#) and [Write](#) statements instead. [Set_Status/Get_Status Error](#) handling should be used when programmatically executing the Copy_Row routine.

See also

[Copy_OS_To_Row](#), [Copy_Row_To_OS](#), [Delete_Row](#)

Example

From the System Monitor:

This command line example shows how to copy the row 5*5 to the key 5*20 in the CAR_PARTS table, overwriting any existing destination row:

```
run Copy_Row "CAR_PARTS", "5*5", "", "5*20", "2"
```

Example 2

```
* Copying Multiple Rows from BASIC+ Program
* Copy rows with keys '1' and '2' from CUSTOMERS table to the CUSTOMERS_BACKUP table.
declare subroutine Copy_Row, Set_Status, FsMsg
declare function Get_Status
row_list = '1': @fm : '2'
Set_Status(0)
Copy_Row( "CUSTOMERS", row_list, "CUSTOMERS_BACKUP", row_list , 0, 0, 0 )
if Get_Status( errCodes ) then
    FsMsg( errCodes )
end
```